FAST PHYSICS-BASED SIMULATION OF VASCULAR SURGERY

WU JICHUAN

(B.ENG HONS, NUS)

A THESIS SUBMITTED
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
DEPARTMENT OF MECHANICAL ENGINEERING
NATIONAL UNIVERSITY OF SINGAPORE

2015

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Jichuan

WU JICHUAN
August 2015

Acknowledgements

Though only my name appears on the cover of this dissertation, a great many people have contributed to its production. I owe my gratitude to all those people who have made this dissertation possible and because of whom my graduate experience has been one that I will cherish forever.

Foremost, I would like to express my sincere gratitude to Prof. Chui Chee Kong and Prof. Teo Chee Leong, my supervisors, for their continuous and enthusiastic support and guidance of my Ph.D study in the past four years. Their patience, motivation, and immense knowledge helped me in all the time of research and writing of this dissertation. They have given me the freedom to explore on my own, and at the same time the guidance to recover when my steps faltered. I could not have imagined having better supervisors and mentors for my Ph.D study.

I would also like to thank Dr. Ooi Oon Cheong from National University Hospital for the cooperation in surgical experiment. I am deeply grateful to him for the long discussions that helped me sort out the technical details of my work.

I would like to acknowledge Dr. Wen Rong, Dr. Xiong Linfei, Dr. Nguyen Phu Binh, Dr. Yang Tao, Dr. Wu Zimei, Mr. Chng Chin Boon, Miss Ho Yick Wai, Yvonne Audrey and many other friends for their invaluable friendship and numerous discussions and comments on related topics that helped me improve my knowledge in the area.

My sincere thanks also goes to Mr. Sakthi, Mrs. Ooi, Ms. Tshin and Mdm. Hamidah in the Control and Mechatronics Lab for their help.

Most importantly, none of this would have been possible without the love and patience of my family. My parents to whom this dissertation is dedicated to, have been a constant source of love, concern, support and strength all these years. I would like to express my heart-felt gratitude to them. My extended family has aided and encouraged me throughout this endeavor.

Summary

Minimally invasive vascular surgery has been shown to be effective for the treatment of vascular or artery diseases. However, extensive training and careful planning are needed for a successful operation due to its inherent problems including non-co-located hand-eye coordination, restricted vision of anatomical environment, and the absence of tactile feedback. Virtual reality medical simulation can potentially provide a safe and robust solution to the training of clinicians.

In order for the training to be effective, the simulation should be physics-based with real-time computer graphics rendering so as to be realistic. This dissertation focuses on the study of fast physics-based simulation of vascular surgery on three important aspects: (1) realistic and interactive simulation of blood vessel deformation using an improved lumped element method, (2) fast hemodynamic simulation in patient-specific model using meshfree method, and (3) simulation of surgical smoke during the surgery.

Human blood vessel can be regarded as a soft-bodied object. A simulation of blood vessel deformation should allow users to interactively manipulate a blood vessel model based on the biomechanical properties of the tissue. The simulation should resemble the soft tissue division process where the blood vessel can be torn into several parts. Therefore, an improved lumped element method is proposed to simulate the blood vessel deformation. The blood vessel model is segmented and reconstructed from clinical CT-images. The method achieves good simulation realism at high computational speed. Real-time simulation and interactive control are demonstrated in an application on patients with abdominal aorta aneurysm by using GPU for general computing.

Study of hemodynamics is normally based on existing cases of patients with specific anomalies. Evaluations and investigations of new cases are limited in practice due to the need for accurate and timely prediction of blood flow. Hemodynamic simulation of blood flow based on the reconstructed patient-specific model is challenging. In this dissertation, two meshfree methods have been used to simulate the blood flow and drug delivery in patient's vascular system. The first approach is based on an improved Smoothed Particle Hydrodynamics (SPH) method which is designed to render a 3D graphical simulation. The second approach is based on an improved Finite Particle Method, and the method was implemented in two cases of blood flow simulations with respect to hand and heart circulation with blockages. The meshfree method avoids the computationally intensive meshing process in conventional hemodynamic simulation. Tensile instability of the SPH formulation is reduced with a smoother approximation of second derivatives in the system. Mass and momentum conservations are achieved locally and globally in the continuum equation. The simulation results have shown to be efficient and realistic for the prediction of blood flow in patient-specific vascular model.

Surgical smoke is generated during the ablation of blood vessel in vascular surgery. It is inevitable that clinicians and patients in operation theatre exposed to the pollution of surgical smoke plume created due to the thermal destruction of tissue. In this dissertation, an improved vortex particle method is used to simulate the dynamics of surgical smoke. The method couples the advantages of both grid-based and particle-based methods. The coarse mesh in the grid-based method is used to predict the main flow trajectory of the smoke fluid, and the vorticity force obtained in the particle-based method is employed to restore the lost details due to numerical dissipations. The realistic and robust simulation can be used for medical training and planning.

Table of Content

List of symbols

∂ : Partial differentiation.

Σ : Summation.

ln : Natural logarithm.

div: Divergence operator: a vector operator that measures the magnitude of a vector field's source or sink at a given point, in terms of a signed scalar.

∇ : Curl operator: a vector operator that describes the infinitesimal rotation of a 3-dimensional vector field.

$\sqrt{a}$: Square root of *a*.

$||$: Vector norm.

$\int$: Integration.

I: Identity matrix.

$min(\cdot)$: Minimum.

$a \times b$: Cross product of *a* and *b*.

$a \cdot b$: Dot product of *a* and *b*.

$a \in b$: element *a* belongs to set *b*.

B^T : The transpose of matrix B.

$\dot{u}$: First order derivative of u.

$\ddot{u}$: Second order derivative of u.

$|x|$: The absolute value of x.

M^{-1} : The inverse of Matrix M.

W_x : Partial differentiation of W with respect to x.

ΔP : The change of P.

$\mathbf{M}$: Mass matrix.

$\mathbb{R}$: Real number.

$\mathbf{B}$: Gradient of deformation vector.

ξ : Strain tensor.

σ : Stress tensor.

f_i : Internal force.

$\mathbf{S}$: Piola-Kirchhoff stress tensor.

f_{ext} : External force.

$\mathbf{T}$: Kinetic energy.

$\mathbf{K}$: Stiffness matrix.

$\mathbf{D}$: Rayleigh damping matrix.

E : Young's modulus.

υ : Poission's ratio (Chapter 4), Kinematic viscosity (Chapter 5,6,and 7).

$\mathbf{G_e}$: Matrix that maps the displacement vector to the strain tensor.

H_e : matrix characteristic of the element e's material properties based on Poisson's ratio and Young's modulus.

$\mathbf{V_e}$: Volume of element e.

$\mathbf{R_e}$: Local rotational matrix of element e.

S : System entropy.

c : Boltzmann constant.

N : Number of particles used in meshfree methods.

$W(|x - x_i|, h)$: Smoothing kernel function.

h : Smoothing length.

$|x - x_i|$ Distance between particles.

ρ : Density.

m_j : Mass of particle j.

A_S : Smooth field.

μ : dynamic viscosity.

v : velocity.

$f_i^{pressure}$: Pressure force.

$f_i^{viscosity}$: Viscous force.

Ω : Computational domain.

$R_n(x-x_i)$: Remaining terms of Taylor's expansion.

$\Delta V_i)$: Lumped volume of particle i.

δ_{ab} : Kronecker delta.

τ : Viscous stress in fluid.

ε : Strain rate in fluid.

p : Pressure.

ς : Void fraction of the fluid.

$I(i)$: All the interior points that are the neighbors of *i*.

$B(i)$: All the boundary points that are the neighbors of *i*.

$G(i)$: All the exterior points that are the neighbors of *i*, in other words, all the ghost particles.

R : Radius.

L : Length.

f_{conf} : Vorticity confinement force.

t : Time.

ω : Vorticity.

List of abbreviations

AAA : Abdominal aortic aneurysm.

API : Application program interface.

BEM : Boundary element method.

CAD : Coronary artery disease.

CPU : Central processing unit.

CFD : Computational fluid dynamics.

CT : Computed tomography.

$CUDA$: Compute Unified Device Architecture.

CVD : Cardiovascular disease.

EVH : Endoscopic vessel harvesting.

FDM : Finite difference method.

FE : Finite element.

$FLIP$: Fluid-implicit-particle.

FMM : Fast Marching Method.

FPM : Finite particle method.

GPU : Graphics processing unit.

GUI : Graphic User Interface.

IHD : Ischemic heart disease.

LIC : Line Integral Convolution.

LV : Left ventricle.

MAC : Marker-and-cell.

MFC : Microsoft Foundation Class.

MIS : Minimally invasive surgery.

MPM : Material point method.

MRI : Magnetic Resonance Imaging.

$N-S$: Navier-Stokes.

NUH : National university hospital.

ODE : Ordinary Differential Equation.

PCD : Peripheral vascular disease.

PDE : Partial differential equation.

PIC : Particle-in-cell.

RA : Right atrium.

RHD : Rheumatic heart disease.

SDK : Software development kit.

SPH : Smoothed particle hydrodynamics.

SOFA : Simulation Open Framework Architecture.

StVK : St. Venant Kirchhoff.

VOF : Volume-of-fluid.

VPM : Vortex particle method.

VR : Virtual reality.

VDSM : Valence Driven Spatial Median.

List of figures

List of tables

Chapter 1

Introduction

1.1 Background and motivation

Medical simulation provides a safe and potentially robust solution for the training and planning of minimally invasive surgery (MIS). Recent development of vascular surgery using indirect operation instruments is an important application of MIS to treat vascular or artery diseases. Artery diseases are induced by plaques or blockages in arteries that transport blood to the patient's heart, limbs and other organs. Pathology studies showed that such diseases could potentially cause atherosclerosis which may result in the narrowing and hardening of artery lumen and increase the blood pressure on the arterial wall. The transportation of oxygen-rich blood to the patient's organs could also be affected due to such etiology [6–8]. Medical reports have shown that Coronary Artery Disease (CAD) which causes inadequate blood flow to the heart is responsible for 2,150 deaths in the United States each day [9].

In traditional open vascular surgery, the operations are conducted on the patient's organs through a surgical incision which is sufficiently large to allow the invasive surgical procedures to take place. The surgery is guided by the clinician's direct visualization. Under such a scenario, it is inevitable that clinicians directly touch the organs and tissues, and expose the patient's in-vivo structure to air. Also, on-site diagnostic analysis requires the clinician to cut through a large amount of healthy muscle layers to reach the parts of interest. These operations could induce many surgical risks including excessive bleeding, pains, infections, and a long healing time due to the large wounds [10, 11].

On the other side of the spectrum, minimally invasive vascular surgery uses specially designed instruments and indirect manipulators to decrease the size of the incisions, and hence, it could reduce injuries to surrounding tissue. Interventional radiology, endoscopic and laparoscopic surgeries are the three mainstream terminologies of MIS. Since the surgery is carried out through a small anatomical opening, it will reduce the surgical risks of traditional open surgery. Post-surgical complications such as internal organ injury, blood clots generation, muscle atrophy, and secondary haemorrhage can also be avoided. Despite the benefits, these surgical procedures share the same limitations that the whole process needs to be guided by indirect visualization through monitors. For example, fluoroscopy and endoscope are commonly used to perform vascular surgery in CathLab [12]. The angle of vision during the surgery is very limited and tactile perception is usually lacking for the clinicians. Therefore, minimally invasive vascular surgery requires very adept operation skill with extensive trainings.

Training and planning of minimally invasive vascular surgery pose several challenges for the surgeons due to its requirements of non co-located hand-eye coordination, restricted vision of anatomical environment, and absence of tactile feedback. Therefore, there is an increasing demand to develop physics-based simulation of blood vessel deformation and blood flow so that the clinicians could acquire those surgical skills in a safe and accurate Virtual Reality (VR) environment. Advancement of medical imaging techniques enables high resolution volumetric data to be captured seamlessly for the assessment of anatomical morphology and pathologic structures. In the past decade, this has become a motivation for a considerable amount of works in the field of surgical simulations [13–16]. It has been demonstrated that the VR-based simulators can lead to a more effective and systematic training, and provide an objective assessment of technical competency [16, 17]. It has also been shown that the skills acquired based on simulation can be transferred into the operation room [18].

1.2 Objective and scope

The objective of this research is to develop a realistic surgical simulation system for training and pretreatment planning of minimally invasive vascular surgery. After reviewing all the potential difficulties in the field, this dissertation study focuses on three most important aspects: (1) realistic and interactive simulation of blood vessel deformation, (2) fast hemodynamic simulation in patient-specific model, and (3) simulation of surgical smoke during the treatment. A detailed survey of relevant literature works and specific gaps will be covered in Chapter 2.

The vascular system of a human can be categorized into three major types: the arteries, which carry the blood away from the heart; the capillaries, which enable the biological exchange of water and chemicals between the blood and the tissues; and the veins, which carry blood from the capillaries back toward the heart [19, 20]. In the sense of solid mechanics, all these blood vessels are termed as soft biological tissues. A simulation of blood vessel deformation should allow users to interactively manipulate a blood vessel model based on the biomechanical properties of tissue. In vascular surgery, clinicians may need to cut some specific blood vessel using surgical tools as in the case of endoscopic vessel harvesting (EVH). Hence, the simulation should also be able to resemble the soft tissue division process where the blood vessel can be torn into several parts when a sufficiently large force is applied. From the perspective of a clinician, an interactive surgical simulation is not ‘convincing’ unless the visualization is performed at a sufficiently high rates. Therefore, algorithm optimization and parallel computation are important so that the simulation can be rendered in real-time.

Diagnostic decisions and treatment planning of modern MIS on artery diseases cannot be evaluated solely by means of static images. Fluidic simulation of blood flow using time-varying medical data is important to understand the vascular circulation under normal and abnormal conditions. The simulation will help clinicians to assess the pathology of diseases, and to compare and evaluate alternative treatment plans. Hemodynamic simulation of blood flow based on the reconstructed patient-specific model, however, is a challenging task and still not fully explored. In this dissertation, the fluidic simulation of blood flow is implemented using meshfree method to avoid the computationally intensive meshing process in conventional hemodynamic simulation. The proposed method is used to evaluate the flow velocity, blood pressure and excessive pressure and force generated on arterial wall to assist the diagnostic for various types of artery diseases.

Coronary artery bypass surgery is a surgical procedure performed to relieve angina and reduce the risk of death from coronary artery disease [21, 22]. Radial arteries from the bifurcation of the brachial artery in the cubital fossa of the patient are harvested by EVH and grafted to the coronary arteries, and they are used to bypass atherosclerotic narrowing and improve the blood supply to the myocardium [23]. During the EVH process, a cautery tool is used to divide tissue with minimal blood loss. As the tissue of the patient is burnt, copious smoke would be generated, which could obstruct the view of the clinician [24]. Such smoke could also lead to severe surgical errors and may result in excessive bleeding and even failure of the surgery [25]. Smoke concentration and increased exposure of the tissues to smoke may elevate the risk of chemical poisoning [26]. Therefore, there is a motivation for developing physics-based simulation of smoke to help surgical training and planning. In this dissertation, an improved Vortex Particle Method (VPM) is proposed to simulate surgical smoke for various initial conditions in patient-specific model. Graphic rendering is also provided to allow clinicians to have an intuitive understanding of the phenomenon.

1.3 Contributions

The contributions of this thesis are as follows:

- A fast and high-fidelity simulation of blood vessel deformation using lumped element method with GPU parallelization;
- A robust, accurate, and patient-specific hemodynamic simulation using meshfree methods and proposed fluidic model;
- A physics-based simulation of surgical smoke in artery during EVH surgery; and
- A software component framework which integrated the design synthesis and architecture of developing physics-based simulation of vascular surgery.

1.4 Simulation of soft tissue deformation

There have been a number of existing works striving to model soft tissue deformation, and they can be generally categorized into three types: (1) Finite-Element-based (FE-based) method, (2) linear-elastic-based method, and (3) other methods based on nonlinear physical processes. As shown in Figure 1.1, the FE-based method are accurate but computationally intensive while methods based on nonlinear physical processes are computationally efficient but lack accuracies [27–29]. The objective of this research in blood vessel deformation focuses on developing a system which balances the simulation realism and computational efficiency, such that the simulation can be timestepped in real-time with an affordable trade-off in accuracy. Regarding these requirements, a method based on the improved lumped element approach is proposed in this dissertation [1]. Unlike the conventional FE method which involves computing of a sparse non-diagonal mass matrix, the proposed method simplifies it into a diagonal matrix using mass lumping. GPU parallelization is then employed so that interactive surgical simulation could take place. The multiple computation units of multi-cores GPU can execute numerous threads in parallel, which is efficient in complex soft tissue simulation [30, 31].

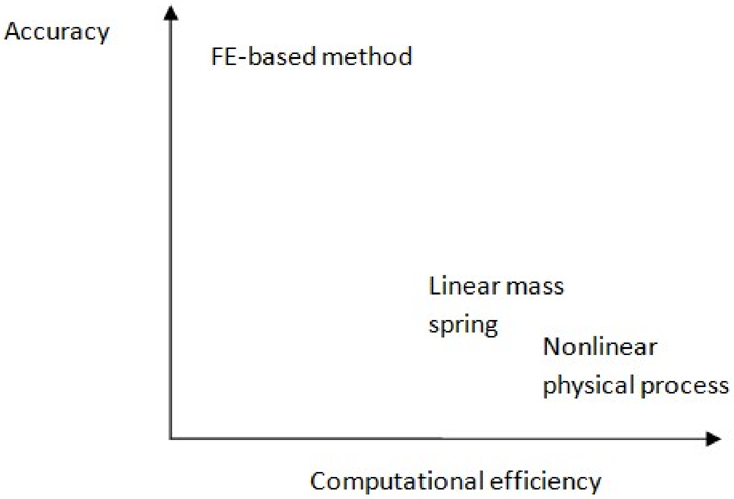


Fig. 1.1 Evaluation of FE-based method, linear-elastic-based method, and other methods based on nonlinear physical processes

1.5 Meshfree methods in hemodynamic simulation

Hemodynamics is the study of blood flow, motion and equilibrium under the action of external forces [32, 33]. As a branch of fluid dynamics in medical application, early researches in this area focus on using mesh-based terminologies developed from highly complex finite element, volume element or boundary element methods[34–36]. Numerical simulations of hemodynamics have been extensively investigated by the fluid dynamics community to allow accurate prediction of blood flow in various types of environment. Nevertheless, numerical simulations are usually valid for only a set of specific and simple geometries due to the computationally intensive meshing process in every timestep. Graphic rendering based on reconstructed patient-specific model are also absent in these works. In this dissertation, an improved Smoothed Particle Hydrodynamics (SPH) method is proposed to simulate the blood circulation of patients with blockages in an artery. The proposed method was based on meshfree framework and a quartic kernel function is used to approximate the function values in Navier-Stokes equations. Tensile instability of the SPH formulation is reduced with a smoother approximation of second derivatives in the system. Mass and momentum conservations were achieved locally and globally in the continuum equation. The results closely resembled physiological behavior of natural human blood circulation. With further development of the algorithm and calibration based on medical experiments, this simulation could be used to assist clinical practises.

1.6 Simulation of surgical smoke

Although numerical simulations of fluidic phenomena such as smoke, fire and water have been extensively studied since 1980s, exploration of these computer-assisted methods in medical application is still lacking. As the advancement of the computer hardware and computational techniques, the theoretical physics community focused on numerical simulations of smoke based on the law of fluid dynamics [37–40]. Their results are generally very accurate, but none of these methods are suitable for real-time medical simulation since they are intended for simulations in a way that each frame is computed over many hours using high-performance computers.

The modeling of smoke and other gaseous phenomena has also received a lot of attention from the computer graphics community over the last two decades. However, their works are usually used in video game industry for visual effects with no dynamic feedback based on the simulation environment, and so their results are generally not physics-based. In this dissertation, a hybrid method is proposed to simulate smoke which couples the advantages of both grid-based and meshfree methods. The method is based on the VPM which uses a large amount of discrete particles to approximate the flow motions of smoke. Each vortex particle carries its own vorticity value throughout the simulation. In order to reduce the intensive computation, the change of vorticity in each particle is defined to be only affected by those surrounding particles and boundary conditions. Conventional mesh-based methods use velocity-stream functions to describe the smoke motion which could result in numerical dissipation and damps out some important flow features. The motivation of VPM was to restore these details so that the smoke in the final rendering could resemble the smoke in the real world [41].

1.7 Thesis organizations

There are eight chapters including this chapter in this thesis:

- Chapter 1 introduces the background, motivation, scope, and contributions of the thesis.

- Chapter 2 reviews the literatures on human vascular system and artery diseases, simulation of soft tissue deformation, simulation of hemodynamics and blood circulation, simulation of surgical smoke, GPU-based parallelization for medical simulation, and the design philosophy of component-based software engineering.

- Chapter 3 describes the proposed component-based modeling framework for blood flow simulation. The framework comprises three main components, namely object model component, image processing component, and GPU acceleration component.

- Chapter 4 presents the simulation method of blood vessel deformation used in this dissertation study. The implementation of the proposed method including reconstruction of patient-specific blood vessel, mesh generation, stress tensor and strain tensor formulation, mathematical modeling of soft tissue deformation, GPU parallelization, and soft tissue division is also discussed in this chapter.

- Chapter 5 introduces the proposed methods used in patient-specific hemodynamic simulation. The proposed methods include an improved SPH method and an improved Finite Particle Method. The derivations for the equations of motion, discretization and boundary conditions are also presented.

- Chapter 6 presents a simulation of patient-specific vascular drug delivery. An experiment of flow regime identification and characterization of the two-phase drug flow are implemented. The simulation results are obtained using the proposed two-phase fluidic model.

- Chapter 7 demonstrates the application of meshfree method in surgical smoke simulation. The implementation includes an illustration of smoke generation during surgical procedures, evaluation of vortex particles, mathematical modeling of topology for smoke particles, and graphic rendering methodologies.

- Chapter 8 concludes the dissertation with a discussion on future work of the vascular surgery simulation.

Chapter 2

Literature Review

Medical simulation is an active research topic focusing on the application of virtual reality-based education and training in medical fields of various industries. After a brief introduction to the human vascular system, this chapter reviews the existing literature on topics related to medical simulation. The main purpose of such application is to train medical professionals to reduce accidents during surgery, prescription, and general practice. The topics of this chapter include human vascular system and diseases, soft tissue deformation, simulation of blood circulation, smoke simulation and Graphic Processing Unit (GPU)-based parallelization for medical simulation.

2.1 Human vascular system and diseases

As an important organ system in human body, vascular system plays an indispensable role in people's health which enables blood circulation and transportation of nutrients, oxygen, carbon dioxide, hormones and other necessary chemicals with regard to the sustainability of the human body systems. Blood cells exchange through the vascular system is a vital procedure for human body to fight diseases, stabilize body temperature and pH, and to maintain homeostasis.

Anatomy study has revealed that the circulatory system is composed of two main components: the cardiovascular system which distributes blood, and the lymphatic system which circulates lymph [42]. If we look deep into the functionalities of these two systems, we should be able to know that they are responsible for most of the substance exchange and protection of fluid environment in human body [43]. The cardiovascular system comprises the blood, heart, and blood vessels, while the lymphatic system is formed by lymph fluid, lymph nodes, and lymph vessels. Blood and lymph are different in the sense of both fluid mechanics and biology. Blood is a fluid consisting of plasma, red blood cells, white blood cells, and platelets that is circulated by the heart through the vertebrate vascular system, carrying oxygen and nutrients to exchange waste material with body tissues. On the other hand, lymph is essentially the recycled excess blood plasma after it has been filtered from the interstitial fluid (between cells) and returned to the lymphatic system [44].

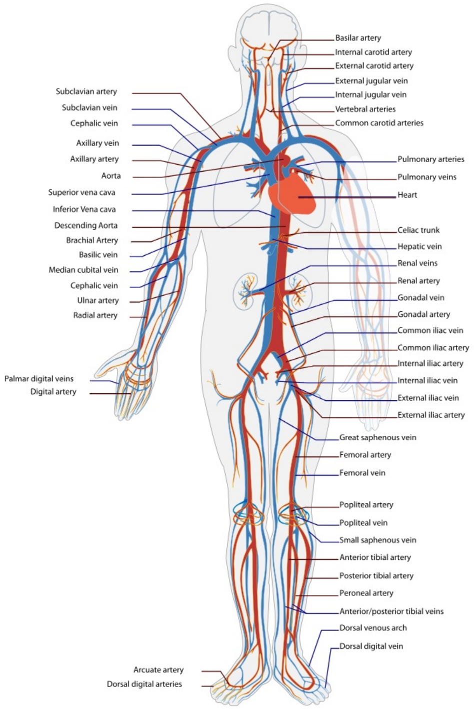


Fig. 2.1 Human vascular system. Retrieved from wikipedia free pictures.

Similar to other vertebrates, humans also have a relatively closed cardiovascular system. The blood fluid circulating in the system never leaves the network of arteries, veins and capillaries. The amount of blood in an adult varies from 4.7 to 5.7 liters which accounts for approximately 7% of the total body weight [45]. As it can be seen in Figure 2.1, there are three major types of blood vessels: the arteries, which carry the blood away from the heart; the capillaries, which enable the actual exchange of water and chemicals between the blood and the tissues; and the veins, which carry blood from the capillaries back toward the heart.

Cardiovascular disease (CVD) is a class of diseases that involve the heart or blood vessels [46]. Common CVDs include: ischemic heart disease (IHD), stroke, hypertensive heart disease, rheumatic heart disease (RHD), aortic aneurysms, cardiomyopathy, atrial fibrillation, congenital heart disease, endocarditis, and peripheral vascular disease (PCD). These CVDs have different pathology and etiology: IHD, stroke, and PCD are caused by atherosclerosis due to high blood pressure, smoking and diabetes, while the etiology of RHD is untreated streptococcal infections of the throat [47–50].

Surgical procedures for CVDs include percutaneous coronary intervention, coronary artery bypass, cutting and harvesting of blood vessel, ablation of spare tissue, implantation of scaffold, screening, vascular drug delivery and controlling of blood pressure. In the past, diagnostic decisions of CVDs' surgery were usually made based on the clinician's experience and direct observation through open surgery [51]. These surgical skills can only be acquired by practicing on human biological specimens and numbers of treatments in operation room. In that case, surgical risks are unavoidable and the possibility of post-surgical complications could be increased due to unskilled operation.

The development of modern computer graphics and medical simulation technology enable the training and planning of CVDs' surgery to be carried out in a safe and efficient manner. For example, soft tissue simulation based on experimental results of human tissue measuring can be used in simulating vascular deformation under different blood pressures;

hemodynamic simulation based on reconstructed vascular system of patient could be applied to evaluate and predict the blood flow and drug delivery in various medical conditions; smoke simulation in an virtual reality environment is able to resemble the tissue ablation procedure and control the amount of smoke generated in the treatment. In this dissertation, these three topics are the main focuses in the physics-based simulation of vascular surgery.

2.2 Soft tissue deformation

Simulation of the deformation of the soft-bodied object's has been extensively studied by the solid mechanics community for decades. The primary interest of this group is to accurately model the deformation of soft body in an off-line and non-interactive manner. Some of their results converge well with the biomechanical properties of simulated soft-bodied object by using mainframe computer for general computing. In medical applications though, it is important that the simulation of blood vessel deformation be interactive. Due to the nonlinear law of physics for the modeling of soft body deformation, algorithm optimization and computation parallelization are indispensable to render the simulation in real time. In this section, I review some representative simulation methods in this area. Based on their methodologies, these works can be categorized as Finite Element-based Method, Linear-elastic-based Method, and other methods based on nonlinear physical processes.

2.2.1 Finite element method

Finite Element (FE) method is the most commonly used approach to model deformable object in solid mechanics [29]. This method decomposes a soft-bodied object into a number of volume elements or boundary elements, and use continuum mechanics model to simulate the deformation under external loads. The philosophy behind this method is to subdivide the problem domain into simpler case, and then evaluate the topological relationship between each subdivision and assemble the final result.

Technically, the Partial Differential Equations (PDE) which describe a soft body can simulate both static response and deformation dynamics. However, static simulation only gives the rest deformation under given external forces, while dynamic simulation integrates the simulated object's mass inertia and damping, and thus, is able to simulate the transient response of the deformation and evolve the simulation based on time-varying parameter updating [52–54].

Mathematically, there are no closed-form solutions for these PDEs except in a few simple geometry cases such as cube, sphere and cylinder [55–57]. In order to timestep the simulation domain numerically, the PDEs are often converted and discretized into Ordinary Differential Equations (ODEs). A commonly used converting method is by inserting the interpolation functions (also called shape functions) into the partial differential equation which describes the motion of deformation [56]. Once a system of ODEs is obtained, the unknown parameters can be transformed into the displacements of the vertices in the meshed model. However, such an implementation poses difficulties in that an ODEs system based on traditional FE method is generally coupled, nonlinear, sparse, and high-dimensional for complex 3D meshes. This limitation caused the most important contradiction in FE method. Conventional FE method is able to render high quality simulation with sufficiently fine meshes; fine meshes, however, induces intensive computational cost to solve the system of equations.

Optimization of FE method is dedicated to enhance the computational efficiency so that it could be implemented in real-time simulation. Among all the difficulties, the most challenging problem is the nonlinear nature of the combined elastic and plastic deformation of soft-bodied objects. Many researchers in this field only evaluated the simplest variants such as linear shape function to describe this phenomenon [58]. In the remaining section, I introduce some mainstream, and reasonably effective, FE-based methods used in real-time soft body simulation. Off-line simulation is not further discussed. However, readers who are interested can refer to [55, 59, 60], which introduced some efficient optimization algorithms such as explicit-FEM and implicit-FEM to reduce the computational cost.

Multi-resolution method is first proposed in [61] and [27]. In order to reduce the intensive computational cost, this method constructs a spatially multi-resolution deformation basis to describe the simulated object. At runtime, the defined basis will dynamically grow or shrink based on the given deformation activity. One thing that should be noted is that the multi-resolution method is purely defined based on geometry with no regards to the spatial distribution of the material parameters.

In the implementation, coarse mesh is employed in most part of the simulated object except the area of deformation which is rendered with fine meshes. The weight of fine mesh used in the system determines the level of computational cost. The accuracy of the simulation could be increased if the whole model is computed using fine meshes. However, if the model is resolved to the finest level everywhere, the advantage of using multi-resolution method would be eliminated and the simulation would be no difference from the standard FE method.

When a pre-computed coarse shape function is used, the object under deformation behaves as if all the meshes in the area of interest are filled up with ‘virtual’ materials. The shape function is also used to describe the biomechanical properties of the simulated object. Some geometric details such as rotation and damping are neglected in the region where coarse mesh is employed. The application of this idea in surgical simulation is, for example to define, for a given anatomical structure, different FE meshes, from a very coarse one to a fully refined one. If the boundary conditions induce small deformations inside the structure, the coarse mesh is sufficient for providing accurate FE discretization. On the other hand, a dense mesh needs to be used where the deformation is high.

Recently, some researchers proposed to use hierarchical deformation bases to adaptively refine the analysis of multi-resolution method [62]. This approach is realized by pre-computing some specific deformation activities with respect to the defined model, such that the accuracy of local deformation can be enhanced. Some methods merged multi-resolution descriptions with Hierarchical FE integrations [53]. Theoretically, such an implementation will be able to provide a numerical scheme which is able to simulate any type of biomechanical description if the defined scheme has gathered enough empirical data to improve the propagation of deformations as well as the efficiency of the computing.

Assumed-shape method is generally a low dimensional method used in computer graphics to simulate deformable object [63–65]. The method approximates a complicated model with a simple geometry which has analytical solution for the continuum equation. This simple geometry is called the assumed-shape, and it is inserted into the center of the problem domain. Given the geometrical model of an object to simulate, the method first compute a bounding box and then recursively subdivide it where needed. The cells of this octree structure are labelled with mechanical properties based on material parameters and fill rate.

At runtime, the surface of the original meshed model deforms according to the assumed-shape geometry [66]. Since the analytical solution has already been solved in advance, computational efficiency is improved and the simulation is able to be timestepped in real time.

The assumed-shape method is commonly used in commercial video game industry which requires fast response but relatively low accuracy in soft body simulation. Some physics engines are also developed based on this method such as the NVidia PhysX and Simulation Open Framework Architecture (SOFA) [67]. Current studies in this area focus on developing variational shape function to refine the physics law which describes object deformation so that higher accuracy could be achieved without much trade-offs in computational cost. Figure 2.2 shows one of my simulations of soft-bodied frogs using assumed-shape method with PhysX. Seven cylinders are inserted to approximate the shape of a frog. The deformation process is simulated according to the defined scheme of each cylinder. Technically, real-time and realistic simulation has been achieved in this work. However, the accuracy is compromised since the shape function which used to describe the deformation activity is not based on the actual biomechanical properties of the simulated soft tissue.

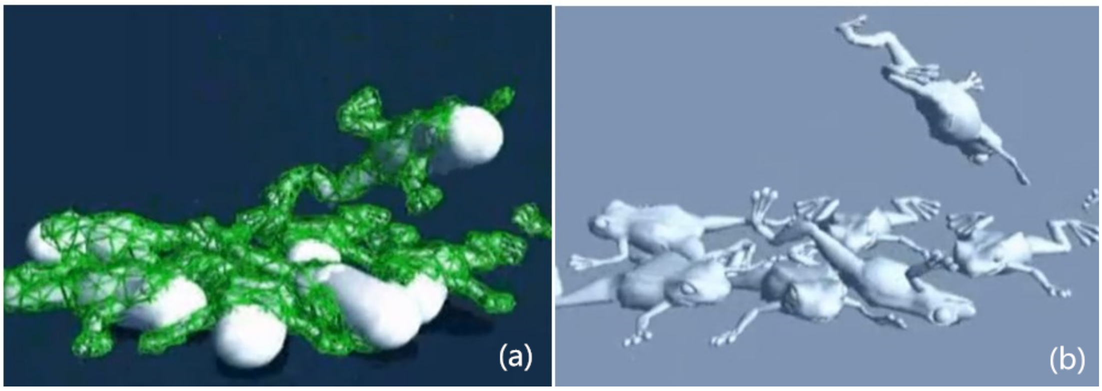


Fig. 2.2 Simulation of soft-bodied frogs using assumed-shape method (the mesh model of frog is provided by PhysX engine)

Modal warping method is a newly developed method by the computer graphics community used in surgical simulation of soft-bodied organs [68]. This is the first method which addresses the nonlinear (large) deformation of soft body in real-time simulation. The method defines the deformation of an object into two modes: linear and nonlinear, which correspond to the elastic and plastic deformation of the material. It starts with pre-computing of the rotational and translational matrices for each element considering a certain range of external loads. At runtime, the pre-element rotations and translations of modal analysis are simply extrapolated to produce fast parametric nonlinear shape model [68–70]. This approach is useful for eliminating gross distortions associated with traditional linear modal analysis.

A very effective point of the method is that the choice of whether the linear or nonlinear shape function should be used only depends on the absolute value of detected force. A step kernel function is employed to determine the threshold of when plastic deformation should happen. Nevertheless, such implementation is also regarded as the limitation of model warping method because once an initial condition is excited, the program will only execute one mode of the shape function and generate single-mode deformation motion [68].

In the solid mechanics community, this phenomenon is also called the nonlinear coupling of modes in soft body simulation. Surgical simulation requires a more generic representation to comprehend the deformation activities and dynamic shape updating on the simulated object. Hence, the modal warping method is normally not used in this field.

2.2.2 Linear-elastic method

Linear-elastic method is also known as linear-mass-spring method. The method decomposes a model into nodes connected by springs and dampers in series or parallel combinations (Figure 2.3). Ease of implementation and generally inexpensive computational cost are the main advantages of the method. On the other side of the spectrum, it is also applicable to simulate the soft body division problem since the nodes used to approximate the shape of the simulated object are connected with virtual springs and dampers. A shape function could be used to define the threshold of force limit upon these connections. The connections would be eliminated when excessive load applying on it is detected to create the effect of soft body division [29, 71, 72].

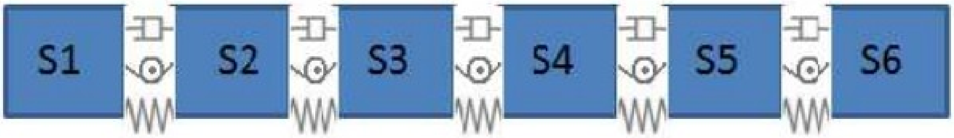


Fig. 2.3 Topological relationship of linear-elastic representation in soft body simulation [1].

In the past two decades, there are many researches dedicated to using medical data and specially designed experiment to measure the physics law of soft tissue in deformation activity. Some well-known results of blood vessel modeling can be found in [73, 74]. Based on the obtained biomechanical properties, physics-based simulation of soft tissue using linear-mass-spring method can be achieved. Some works try to lump the mass of the model on each meshed segment (usually in square or triangle shape) [27]. However, such methods pose two challenges which are contradictory to each other. First, if the mesh is not fine enough, the rendered graphics would look inauthentic since the object is just formed by pieces of ‘squares’ or ‘triangles’. This phenomenon is more obvious when the simulated object is under a rotational load and twisted. Second, if the simulation employs fine meshes to approximate the shape of the soft body, the computational cost would become very intensive. Realistic rendering of the deformation activity requires very huge amount of meshed segments to deceive the perception of human eyes, the same problem in FE-based method.

Figure 2.4 shows one of my early works of simulating a single layer soft tissue using the linear-elastic method. Interactive simulation is achieved in this work which allows the user to manipulate model deformation when external forces are applied on the simulated object. Some basic features of soft tissue such as linear deformation and twisting with respect to the biomechanical properties retrieved from medical report are achieved and rendered in an effective manner. Optimization of the model geometry enables the simulation to be processed using a typical personal workstation. However, as can be seen from the result, the simulation does not resemble the natural soft tissue at a sufficiently fine level. Hence, the work is not suitable to be used in medical applications.

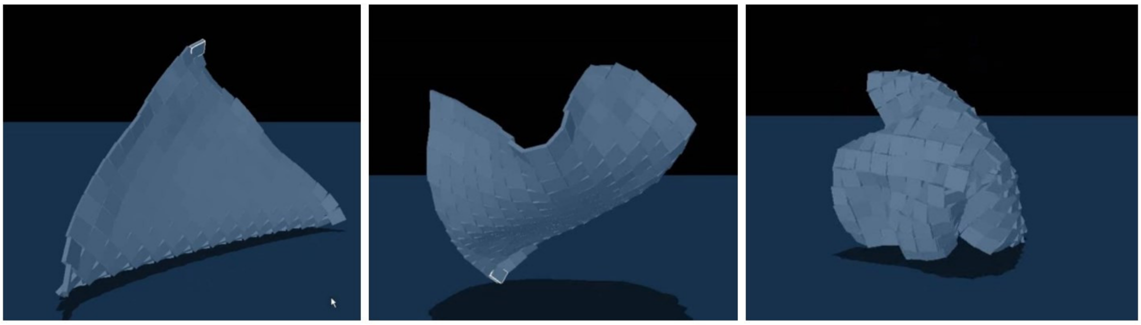


Fig. 2.4 Simulation of soft tissue using linear-elastic method

Linear-mass-spring method is adequate to simulate small (linear) translational deformation of soft body. Nevertheless, large (nonlinear) deformation must also be included in surgical simulation due to the material nonlinearity of soft tissue. Yang in [73] revealed the nonlinear stress-strain relationship of blood vessel in both longitudinal and circumferential directions. The original mass-spring model can be improved by introducing two groups of material constants (two spring coefficients and two damping ratios) to control the deformation mechanism [71]. Such modification could increase the degrees of freedom for control parameters. However, it does not comply with the constitutive law of the simulated object since the proposed material parameters are arbitrarily assigned. Fixed deformation coefficients are used through the whole simulation with no dynamic updating with respect to the actual deformation law of the material. Surgical simulation of blood vessel deformation generally demands higher realism and more accurate solutions.

2.2.3 Other methods based on nonlinear physical processes

In recent years, some researchers have proposed a novel concept of using nonlinear physical processes to simulate soft tissue deformation in real time. This concept is inspired by the fact that the analytical solutions for many physical processes can be easily obtained in advance, and thus, the cost of computation can be largely reduced by using such a physical process to approximate the description of deformation activity for a soft-bodied object [75].

There are several models which are commonly employed by researchers using this methodology including Poisson's equation, auto waves, cellular neural network, ideal gas model and reaction-diffusion model. In their works, the complex nonlinear deformation problem is converted into a simple nonlinear energy propagation problem, which avoids complex elastic/plastic computations of mesh displacement [28, 76–78]. Direct application of the model onto the simulated object would be inaccurate. However, the simulation results can be calibrated with respect to biomechanical data so that the output deformation shape function is able to converge with the physics law in a certain range.

Theoretically, all nonlinear physical processes can be adopted as the governing law for soft tissue simulation in either surface or volume rendering. In this section, two sophisticated and representative models are introduced: the ideal gas model and the reaction-diffusion model.

Ideal gas model is probably the first method of using nonlinear physical process to simulate soft tissue deformation. It was proposed in a SIGGRAPH paper in 2004 [28]. The author converted the complicated soft body simulation to a simple thermo-dynamics model and used Clausius-Clapeyron state equation for the modeling of pressure distribution. A force accumulator was used to store and update the solutions of the state equation. A customizable spring mass engine integrated with Newton's Second law was developed to evaluate the physical representation of the force accumulator. The engine basically maps the deformation function onto the whole problem domain, and the simulated soft body is able to deform when a fixed or variational pressure function is applied.

It has been shown that this method is very effective in simulating 'balloon' type of soft-bodied object. Many of the soft monsters appearing in video games are simulated using the method. In such scenario, surface materials are homogeneously distributed. The propagation and deformation due to external load can be rendered realistically based on the ideal gas model.

Reaction-diffusion model received attention from the surgical simulation community due to its relatively simple structure in simulating soft-bodied object. In 2006, the theoretical basis and potential of using reaction-diffusion model for soft tissue simulation was first discussed by Muraki and Kita in a survey of medical simulation techniques [76]. Sermesant et al. developed a pro-active deformable heart model by using reaction-diffusion model [77]. However, this work only simulated the internal interactions of the model with no contact with other objects. In 2012, Zhong et al. [75] calibrated the model based on the theoretical analysis of soft tissue behaviors from the physicochemical viewpoint. Different from the conventional methodology of using analogy drawing to describe the observations of physical phenomena, Zhong's work is more accurate since the model is established on existing physics law of reaction-diffusion and the dissipation flow of Navier-Stokes equations.

The limitation of using nonlinear physical process to simulate the deformation activity is that the method is only capable of describing the macroscopic level effects of the soft-bodied object. Experimental calibration could be used to enhance the simulation accuracy. Nevertheless, the result could converge with its natural counter-part in the real world only in a very small region of the problem domain. In medical applications, nonlinear physical process models are constrained to simulate elastic-deformation with a relatively steady input of external loads.

On the other side of the spectrum, the nonlinear physical process can still be used to simulate some interesting feature of soft body as long as the implementation is not mapped onto the whole problem domain. In this dissertation, an exploratory algorithm based on entropy threshold is used to simulate the division of soft tissue in blood vessel. Entropy is a physics term which is normally used to measure the 'disorder' of a system in energy releasing process. More details of the implementation will be introduced in Chapter 4.

2.3 Simulation of blood circulation and hemodynamics

Hemodynamic simulation of blood flow is important for understanding the pathology of many human diseases. There is an increasingly demand to develop accurate and robust simulation of blood circulation based on the law of fluid dynamics. There are many methods developed by the fluid dynamics community to simulate various types of liquid flow in different geometries. However, simulation of blood flow based on the reconstructed patient-specific model is a challenging task and still lacks exploration.

Simulation of fluid dynamics can be defined in a way that by giving the initial conditions and scene geometry, the simulator evolves the motion of the fluid forward in time based on Navier-Stokes (N-S) equations (or some simplified forms) [34]. The N-S equations are the governing laws which describe the physics of fluids. Early researches in the field were carried out by the physics community, who has derived a variety of mathematical equations to describe the dynamics of fluids since 1950s. Their contributions form the basis of today's advancement of fluid simulation used in many applications. In computer graphics, such applications range in complexity from extremely time-consuming high quality animations for 3D films' visual effect, to simple particle systems used in commercial video games [79, 80].

The work contributed by T3 group at Los Alamos National Laboratory in the United States is regarded a milestone in the field back in the 1960s [2, 81, 82]. This group did quite a number of pilot studies and invented many effective techniques, including the famous staggered Marker-And-Cell (MAC) grid structure and the Particle-In-Cell (PIC) method. These methods are the foundation of today's widely used Fluid-Implicit-Particle (FLIP) method, Material Point Method (MPM), and other hybrid methods [4, 83–85].

Conventional methods of simulating fluid in human artery can be classified into two branches, the Lagrangian-based method, where fluid evolves in moving frame, and Eulerian-based method, where fluid evolves in stationary frame. These methods are timestepped on top of a grid-based geometry [86]. In multiple-phases fluidic simulations, the Eulerian method treats the particle phase as a continuum and develops its conservation equations on a control volume basis which is similar to the form of fluid phase. Different from that, the Lagrangian method considers particles as a discrete phase and tracks the pathway of each individual particle. By studying the statistics of particle trajectories, the Lagrangian method is also able to calculate the particle concentration and other phases' data. In recent years, meshfree methods are also proposed to simulate fluids with non-constrained boundary conditions. In this section, the literature reviews of these three types of methods and the simulation of blood circulation are presented.

2.3.1 Eulerian-based method

Compared with the Lagrangian-based method which tracks the motion of particles in a moving frame, the Eulerian-based method looks at fixed points in space and computes the fluid quantities (such as density, velocity, temperature, etc.) measured at those points in a time-varying system. When the fluid is flowing past those points, the change of the corresponding quantities can be derived with respect to the defined stationary frame [87]. This mechanism can be explained using an example of a warm fluid moving past followed by a cold fluid. In this process, the temperature at a fixed point in space would decrease even though the temperature of any individual particle in the fluid would not be changed.

The **Marker-And-Cell (MAC)** is the most famous Eulerian-based method first developed by Francis Harlow in 1965 [88]. In 1970, Amsden et.al [2] developed a simplified MAC method (SMAC) which circumvented difficulties with the original method by splitting the calculation cycle into two parts, where a provisional velocity field calculation followed by a velocity revision to ensure incompressibility throughout. Due to the limitation on the size of time step in conventional MAC method, Deville [89], Pracht [90] and Golafshani [91] developed an implicit scheme to overcome this problem [3]. Pracht's work is basically an implicit treatment of the velocity which is similar to the implicit-fluid Eulerian method. Such implementation allows the simulation of flows involving curved or moving boundaries.

The MAC method consists of two components: markers, which define the location of fluid in a particular timestep, and grid cells, which store the physical parameters that control the motion of fluid. The marker is generally a particle system which describes the fluid motion at a finer level than the traditional spatial grid, which is regarded as relatively coarse in fluid dynamics [88].

MAC is essentially a convention used to represent fluidic data on a grid in an Eulerian representation. In a staggered MAC grid, the method only gives one vector per grid cell instead of three vectors, and each component is stored on the cell faces. In a non-staggered

MAC gird, the system only assigns one single vector per cell, and all fluidic components are stored at the cell center. Figure 2.5 shows the concept of a typical MAC grid [92].

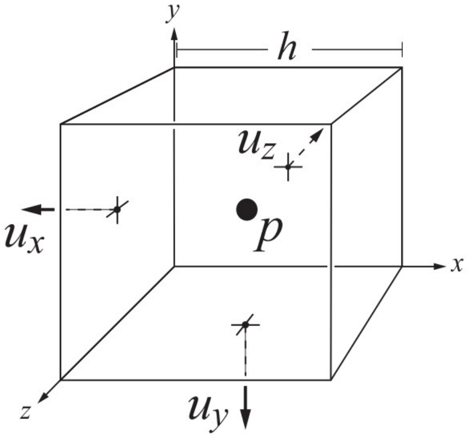


Fig. 2.5 A MAC grid cell. Velocity components, u_x, u_y and u_z, are stored on the minimal faces of the cell. Pressure, p, is stored at the cell center [2].

Early applications of MAC method in computer graphics subdivide the fluid into a set of equally shaped cubes (Figure 2.6). In order to refine the simulation, some researchers proposed the model of using cubes at the surface and near the bottom of the fluid geometry, and long tall cells in the middle [93]. This implementation is not a big advancement in fluid dynamics, but it demonstrates the concept of reducing computation while keeping the important surface accuracy and fidelity. The concept has been widely adopted in optimization algorithms of today's surgical simulation.

2.3.2 Lagrangian-based method

Lagrangian-based simulations of fluid dynamics are usually divided into two branches, the semi-lagrangian framework such as the Particle-In-Cell (PIC) method, and the full-Lagrangian framework such as the Fluid-Implicit-Particle (FLIP) method.

PIC is a mainstream method used in early day's fluid simulation. The method organizes and analyzes the fluid system as groups of particles in cells. At runtime, the motion of cells

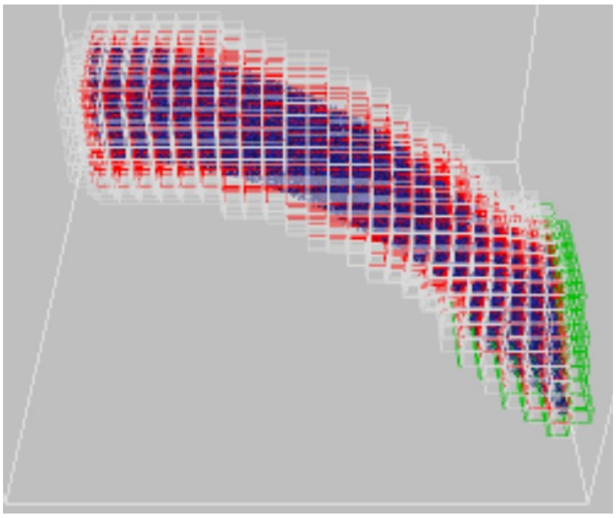


Fig. 2.6 Graphic representation of fluid simulated using MAC method [3].

affecting the grid is solved by the Navier-Stokes (N-S) equations, and then the program moves the cells based on the solution and transfers their vector information. This process is repeated recursively during simulation [81, 82]. One thing which should be noticed is that the method does not evaluate each particle individually. Instead, it evaluates the moving cells (also called the particle mover) in the system. Mathematically, PIC is a hybrid system of Lagrangian and Eulerian method. Particles of the fluid are evaluated in a Lagrangian frame (moving frame) and tracked in continuous phase space, while moments of the distribution are computed in the Eulerian frame (stationary frame). PIC was once the only method which was able to describe interactive/distorted fluid in two or three dimensions, for which it receives considerable attentions to be used in many types of complex fluid simulations [94].

FLIP method is extended based on PIC. In mathematics, PIC is considered as noisy, and it has more numerical viscosity and heat conduction than are acceptable by high-fidelity 3D simulations. Numerical diffusion caused by the continuous averaging and interpolating of the fluid variables is the essential problem which makes traditional PIC method become historical interest [4].

FLIP method addresses the problems of PIC by using a full Lagrangian representation of particles so that the convective transport, which is the main contribution in computational diffusion, is eliminated. In FLIP method, the Lagrangian particles contain all parameters that are used to describe the fluid. At each time step, the values of these parameters are updated by solving the moment equations derived from N-S equations [95, 96].

In recent years, some works dedicated to using a combination of PIC and FLIP method to simulate more complicated feature of fluid in 3D, so that the advantages of both Lagrangian and Eulerian system can be included (Figure 2.6). A remarkable work of using this hybrid method is presented in Zhu and Bridson's SIGGRAPH paper 'Animating Sand as a Fluid' [97]. They introduced a novel approach of combining the strengths of particles-based and grid-based method. The core representation of fluid using this method is a cloud of particles, which allows for accurate and flexible surface tracking and advection. Besides, they use an auxiliary grid to define the boundary conditions and incompressibility. This work demonstrated the basic concept of how to simulate multi-phase objects as fluid.

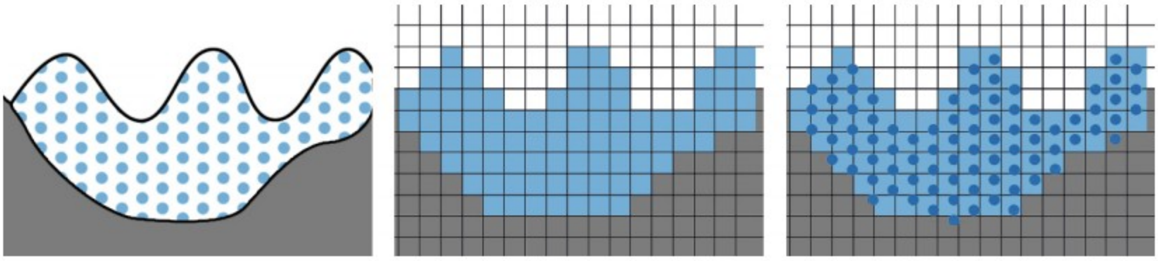


Fig. 2.7 A Lagrangian, Eulerian and Hybrid viewpoint of fluid respectively [4].

2.3.3 Methods based on meshfree frameworks

Recent developments in meshfree and particle methods provide alternatives for traditional numerical methods in modeling fluid dynamics. Among these meshfree methods, smoothed particle hydrodynamics (SPH) is unique and has some obvious advantages. The SPH method, first proposed in 1977 by Lucy, Gingold and Monaghan, is an explicit meshfree, free Lagrangian, particle method used in computational fluid simulation [98–100]. The method divided the fluid into a set of discrete Lagrangian particles. Each particle carries its time-constant mass and time-dependent fluid field equation based on the N-S equations. The motion of particles described by this system are only affected by other particles within a certain range called smoothing length. Beyond this range, the interactions from other particles are 'smoothed' by the kernel function $W(r)$. This method is very computationally efficient because the mass of each particle is automatically conserved, and the complicated convective acceleration term in N-S equation can be replaced by a simple differentiation of particle in time domain.

SPH has been studied and applied in practical applications in recent years. In [101], SPH was used to model incompressible, free-surface hydrodynamic flows, and it decoupled the pressure term from the velocity field by introducing an artificial sound speed and an artificial equation of state. Becker and Teschner reduced the problem of fluid compressibility with stiffer Tait equation, but their simulation was only implemented with simple geometry [102]. Muller et. al seeded ghost particles in the model to simulate the boundary conditions [103], however, the interaction between flow and boundary object is still lacking. Vazquez-Quesada and Ellero [104] simulated the flow of Oldroyd-B liquid around a linear array of cylinders confined in a channel and compared the dimensionless drag force acting on the cylinder with the available results. Hashemi et al. [105] studied the movement of suspended solid bodies in fluid flows using an explicit weakly compressible SPH method. The research works mentioned above are mainly considered in general shape instead of patient-specific models, and this might be due to the absence of medical data and reconstruction method.

2.3.4 Simulation of blood circulation

Due to the ever-increasing interest on blood flow dynamics, many researchers have conducted theoretical, experimental and numerical simulation works in this area. Theoretical researches are usually valid for simple cases with linear fluid dynamics of blood flow in various geometries. Experimental works are generally expensive and sometimes certain physical phenomena and parameters cannot be scaled in a practical experimental setup. Recently, an increasing amount of researches focus on numerical simulations with the advancement of the computer hardware and computational techniques. Some of these works include the visual simulation of PMMA injection during vertebroplasty [19], and local delivery of chemotherapy in vascular system during chemoembolization [106].

There are other works dedicated to simulate the hemodynamics for different types of geometries, such as [107–110]. Jung and his team proposed a multiphase transient non-Newtonian three-dimensional computational fluid dynamics (CFD) simulation to evaluate the pulsatile hemodynamics in an idealized curved section of a human coronary artery [111]. Chaniotis and his team performed a computational study of pulsatile blood flow in prototype vessel geometries of coronary segments. Their study analyzed the wall shear stress (WSS) spatio-temporal distributions for non-Newtonian blood flow in two different conditions including curved pipes model with different curvatures and bifurcating pipes model with different branching angles and flow division [112]. Nevertheless, a common limitation of these works is that their simulations are not based on patient-specific models.

Most of the hemodynamic simulations are implemented using grid-based methods, such as Finite Difference Method (FDM), Finite Element Method (FEM), and Boundary Element Method (BEM). As a complex fluid motion, blood flow circulation involves changing and breakup of free surfaces, complicated streamline forming, fluid-solid interaction, boundary pressure analysis and conservation of mass and momentum. It is therefore difficult for traditional grid-based numerical methods to model such complicated physical processes. For example, traditional FEM cannot treat large fluid deformation very well, and mesh adjustment or rezoning can be necessary in model liquid-solid interactions. Traditional FDM requires special algorithms such as volume-of-fluid (VOF) [113] and Level Set [114]] methods to track changing free surfaces or moving interfaces, which are computationally intensive [86].

2.4 Smoke simulation

Early researches of smoke simulation were done by the theoretical physics community focusing on numerical simulations of smoke by employing the constitutional law of fluid dynamics [39, 40, 115, 116]. Their results are generally very accurate. However, none of these methods are suitable for real-time simulation since they are designed for simulations in which each frame is computed over many hours using high-performance computers.

The modeling of smoke and other gaseous phenomena has also received a lot of attention from the computer graphics community over the last two decades. Some models focused on specific phenomena and animated the smoke's density directly without modeling its velocity [117–119]. Additional details were added using solid textures whose parameters were animated over time. Subsequently, random velocity fields based on a Kolmogoroff spectrum were used to model the complex motion characteristic of smoke [120]. The limitations shared by all these models are that they had no dynamic feedback based on the simulation environment and their results are generally not physics-based.

A more natural approach to model the motion of smoke is to simulate the equations of fluid dynamics directly. Kajiya and his team were the first in computer graphics community to implement this concept [121]. Nevertheless, their works only produced results on very coarse grids due to the limitation of computational power at that time. Except for some models specific to two dimensions [122, 123], no notable progress was made in this direction until the work of Foster and his team who designed simulations of using relatively coarse grids to produce the important swirling smoke motions in three dimensional spaces [124]. Foster's method used an explicit integration scheme to solve the continuum equations, and the simulation is only stable if the time step is chosen to be sufficiently small. Therefore, their simulation is very slow for the simulation of high velocity smoke. In order to alleviate this problem, Stam et.al introduced a model which is unconditionally stable and consequently could be run at any speed [80, 125]. This was achieved using a combination of a semi-

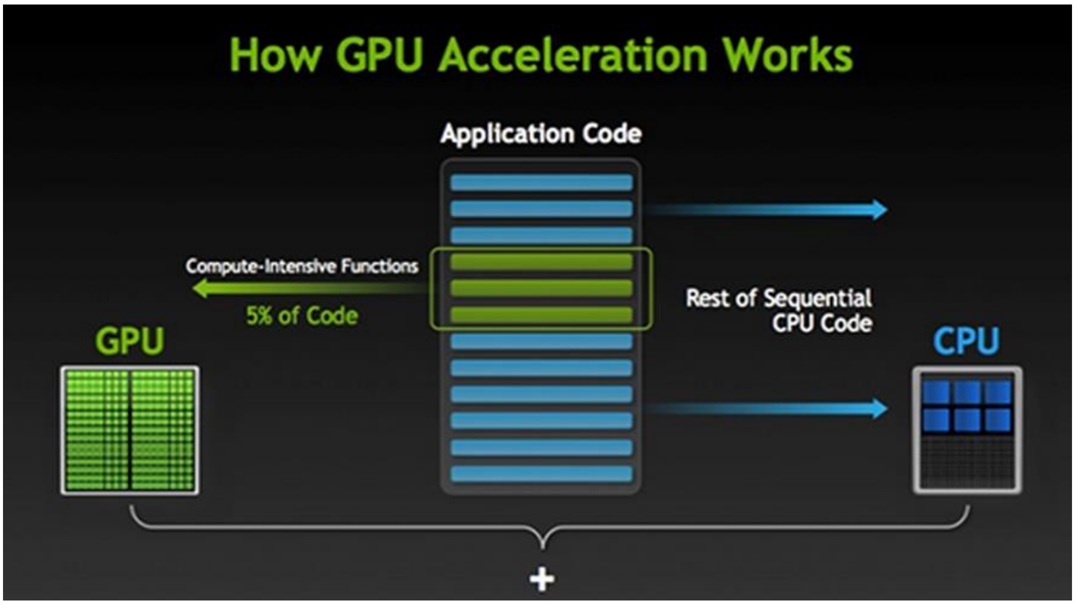
Lagrangian advection schemes and implicit solvers. Nevertheless, a first order integration scheme was used in this model, and the simulations suffered from high numerical dissipation. Although the overall motion looks fluid-like, small scale vortices of smoke vanish rapidly and the details are unable to be restored.

Recently, Yngve et al. proposed a method to solve the compressible version of the equations of fluid flow using model explosions in their SIGGRAPH paper 'Animating explosions' [126]. While the compressible equations are useful for modeling shock waves and other compressible phenomena, they introduce a very strict time step restriction associated with the acoustic waves. Most CFD practitioners avoid this strict condition by using the incompressible equations whenever possible [117]. The compressible flow equations are not considered in this work since the main interest of the proposed smoke simulation is to resemble the tissue ablation procedure and evaluate the amount of smoke generated in the treatment.

Smoke is an incompressible, single-phase and turbulent fluid. Simulation of smoke based on N-S equations contains velocity fields with large spatial deviations associated with turbulent and rotational structures. Conventional mesh-based methods use velocity-stream functions to describe the smoke motion which could result in numerical dissipation and damps out some important flow features. There is another type of simulation method proposed to restore these details so that the smoke in the final rendering could resemble its counter-part in the real world. Such a method is called the Vortex Particle Method (VPM). The method uses the convective force term in the N-S equations to restore the lost details of smoke flow due to numerical dissipation. The implementation of smoke simulation in this dissertation is also developed based on VPM, and more details will be introduced in Chapter 6.

2.5 GPU-based parallelization for medical simulation

Video cards or graphics processing units (GPU) have become tools for processing in parallel large amounts of information. NVIDIA developed the Compute Unified Device Architecture (CUDA) architecture that groups cores of GPU in a vector which can be programmed to reduce processing time for large amounts of data [127]. GPU-accelerated computing is the use of a graphics processing unit (GPU) together with a CPU to accelerate scientific, medical, analytics, engineering, consumer, and enterprise applications. GPU-accelerated computing offers unprecedented application performance by offloading computationally intensive portions of the application to the GPU, while the remainder of the code still runs on the CPU [5, 128, 129]. This combined computing platform is shown in Figure 2.8.



How GPU Acceleration Works

Fig. 2.8 A combined GPU-CPU computing platform [5].

An intuitive way to understand the difference between a CPU and GPU is to compare how they process tasks. As it is shown in Figure 2.9, a CPU consists of a few cores optimized for sequential serial processing while a GPU has a massively parallel architecture consisting of thousands of smaller, more efficient cores designed for handling multiple tasks simultaneously.

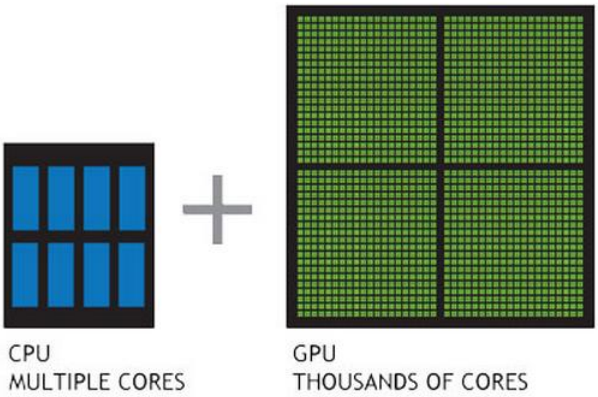


Fig. 2.9 Comparison of CPU and GPU [5].

The use of a GPU to parallelize tasks started in the past decade. For example, Fung and Mann proposed a new architecture using multiple GPUs for image processing and computer vision and they obtained significant speed improvement over a CPU-only implementation [130]. Farrugia and his team developed the GPUCV library to speed up the computational process using GPUs and they observed that processing time with GPUCV varies from 1.2 to 18 times faster than native OpenCV function [131, 132].

In the field of medical simulations, there are some examples which have demonstrated the implementation of non-linear finite element algorithms to compute soft tissue deformation on GPU [30, 31, 60, 128, 133]. The first implementation of the basic total Lagrangian explicit dynamics algorithm on GPU has been demonstrated by Taylor et. al in [134]. Their implementation proved that the algorithm is suited for execution on GPUs and other parallel hardware. The results have shown 16 times computational speed gain compared with the corresponding implementation on a CPU-only system from a typical personal computer. Nevertheless, several limitations have been observed in their work. First, the program is only able to handle linear locking tetrahedral elements and a single material type and contains no features for modelling contact interaction. Second, there is no integration step control included in the implementation, and thus, steady state solution is not able to be computed through this algorithm.

Joldes et al. utilized the NVIDIA CUDA to implement their finite element algorithms for the simulation of soft body deformation [135–137]. These works overcame the limitations of previous implementations as they include dynamic relaxation that facilitates fast convergence to steady state solution, brain-skull contact model, several non-linear materials, and supports hexahedral and non-locking tetrahedral elements that are very efficient and robust in modeling of incompressible continuum such as the human organs and other soft tissues. Wenrong et al. proposed a GPU-acceleration method in Augmented Reality (AR) medical application for parallel matrix operation based on CUDA architecture for accelerating matrix calculation and medical image processing [138]. The sparse matrix in the simulation are parallelized through different kernels and computed by independent processing unit. The result is assembled in the CPU only at the end of computation to reduce the intensive computational cost of CPU-GPU transfer of data [133].

In order to obtain high computation performance, strict guidelines must be followed when programming a GPU conventional SDK such as CUDA and OpenCV. One of the most critical guidelines is the data transfers between the CPU (that runs a program from which a GPU kernel is launched) and GPU. Such transfers are relatively slow and in order to minimize them, in the GPU implementation of soft tissue simulation, all the information needed for the computations is transferred to the GPU in the initialization stage [5, 127]. A sophisticated procedure for such implementation is summarized as follows:

- Compute the damping coefficient, the time step and mass matrix that facilitate the fastest convergence to the steady state;

- Precompute all other needed quantities/variables such as the element shape functions, hourglass shape vectors for under-integrated hexahedral elements, initial volumes of the elements etc.;

- Transfer all the needed data to the GPU memory.

In medical simulations, the computational hardware available in high performance workstations shifted from increasingly efficient but complex sequential computational units, to smaller units, each not much faster than previous generations, but duplicated to be able to execute more threads in parallel. The advantage of multi-cores GPU parallelization has been adopted as an acceleration unit in the physics-based simulation of blood vessel deformation and blood flow in this work.

2.6 Component-based software engineering in physics-based simulation

The advancement of computer graphics and simulation technologies enable the training and planning of artery disease surgery to be carried out in a safe and efficient manner. There are component-based physics engines available for the simulation of fluid in various geometries developed by the computer graphics community. Many of them have conventional API libraries which allow programmers to develop their own projects using pre-defined algorithms.

Nvidia PhysX is a multithreaded physics simulation SDK running on Compute Unified Device Architecture (CUDA). It can simulate volumetric fluid utilizing Nvidia's GPU acceleration techniques. **Bullet** is an open source physics engine which has plug-ins/components for **MAYA** and **Softimage** so that developers with no fluid mechanics knowledge are able to generate fancy visual effect of liquid in their programs. **Blender** engine is a toolkit which simulates the fluidic phenomena using advanced mesh evolving method so that the interaction of fluid with solid object can be pre-computed and rendered in a video. These physics engines focus on advanced rendering and shading techniques to produce fancy graphical fluid simulation for the user of video game. Resembling the natural counter-part in the real world based on physics law is not their primary interest. Simple one-phase laminar flow model based on meshed geometry is the most commonly used approach in these physics engines [127, 139–142]. The newly developed meshfree methods for fluidic simulation are largely absent in these engines.

At the other side of the spectrum, there are highly complex finite element software for fluid simulation developed by the fluid dynamics community. These software are also component-based which allow users to customize their simulations using embedded modulus. Sophisticated Finite Element-based (FE-based) commercial software including **COMSOL** and **Abaqus** have been widely used by engineers and researchers for the study of fluid flow in their designed fluidic structures and geometries. Such software allow display and visual manipulation of the input volumetric data through a standard Windows Graphic User Interface (GUI).

COMSOL has many embedded fluidic modulus including laminar, turbulent, one-phase and multi-phases models readily available for the users to access from the library. By importing the user-defined initial and boundary conditions, users are able to construct their own simulation for a specific fluid [143].

Abaqus is a similar software for finite element analysis and computer-aided engineering applications [144]. It can be used for both modeling and analysis of mechanical components, and visualizing the finite element analysis result. Open-source scripting language **Python** is used for scripting and customization in the toolkit. The limitation of these commercial softwares is that the input geometry is either simple regular shape model or generic model [145]. Importing of patient-specific reconstructed model from clinical CT-images or MRI cannot be meshed properly and the boundary conditions would not be able to be defined in Abaqus and COMSOL. Besides, even though high accuracy can be achieved, the intensive computational cost of FE-based methods usually demand hours of time for the simulation which makes the software inappropriate for on-site surgical planning and training.

2.7 Summary

This chapter reviews the literature related to my PhD research on medical simulation. Tables 2.1 and 2.2 summarize the various methods for soft body deformation and blood flow simulation. There are relatively few works on smoke simulation which is common during surgery. Real-time rendering of complex and computational intensive physics-based medical simulation can be achieved with the utilization of GPU-based parallelization.

Table 2.1 Summary of simulation methods for soft body deformation

Category	Methods	Computational cost	Accuracy	Physics-based
FE-based method	Multi-resolution Assume shape Modal warping	High	High	Yes
Linear-elastic method	Mass-spring Model lumping	Low	Medium	Yes (for elastic deformation)
Methods based on nonlinear Physical processes	Ideal gas Reaction-diffusion Cellular neural network	Medium	Low	No

Table 2.2 Summary of methods for blood flow simulation

Category	Methods	Computational cost	Accuracy	Real-time
Eulerian-based	MAC Implicit-fluid	High	High	No
Lagrangian-based	PIC FLIP	High	High	No
Meshfree method	SPH FPM	Medium	Medium	Yes

Chapter 3

A Component-based Modeling Framework for Blood Flow Simulation

Physics-based modeling for medical applications has been extensively developed, upgraded and customized in recent years. It has been shown that object-oriented programming technique has significantly optimized the implementation efficiency and extensibility as well as demonstrated the advantage of ease of maintenance for complex engineering software [146]. Nevertheless, developers are still required to modify the source code of the program when a potential upgrade is required. This limitation could lead to multiple and even incompatible versions of the same program or application. In this chapter, a component-based approach modeling framework for simulation of blood vessel deformation and blood flow is proposed. Programs developed based on the framework can be easily customized and extended by other researchers and developers. Input and output data updating are also systematically integrated with the support of standard workflow, software ports and interfaces.

3.1 Overview

There are a number of works dedicated to simulating vascular blood flow using different methods. Nevertheless, we have yet to see a systematic software framework which could test different fluidic simulation methods on reconstructed patient-specific model before a specific computational method with the designed accuracy is recommended. There is a trade-off between realism and interactivity in medical simulation. This chapter focuses on the component software development approach of hemodynamic simulation, and detailed implementation of each component will be introduced in subsequent chapters.

Software component is the main feature of component-oriented software development. Each component should have its own implementation procedures for specific object input. According to [147], a software component should possess four properties including encapsulation, descriptive, replaceable and extensible so that it can be extensively developed, upgraded and customized when a different algorithm is used.

Figure 3.1 is an overview of the framework for the proposed software component approach to GPU accelerated physics-based blood flow simulation. The program is built based on Microsoft Foundation Class (MFC) and implemented in C++ language. OpenGL is used for graphic rendering and CUDA toolkit is employed for GPU acceleration. The framework comprises three main components, namely object model component, image processing component, and GPU acceleration component. Each component is designed to be executed independently so that it can be used in different applications software. Access and modification of the component can be easily done based on the component process flow.

The image processing component is designed to reconstruct the raw CT-images into 3D patient-specific models. These models would be used as the input geometry for the simulation of blood flow in the object model components. The object model components combine a FE modeler for solid deformation simulation of the boundary and a meshfree modeler for fluid flow simulation based on an improved SPH method. Such simulations are generally

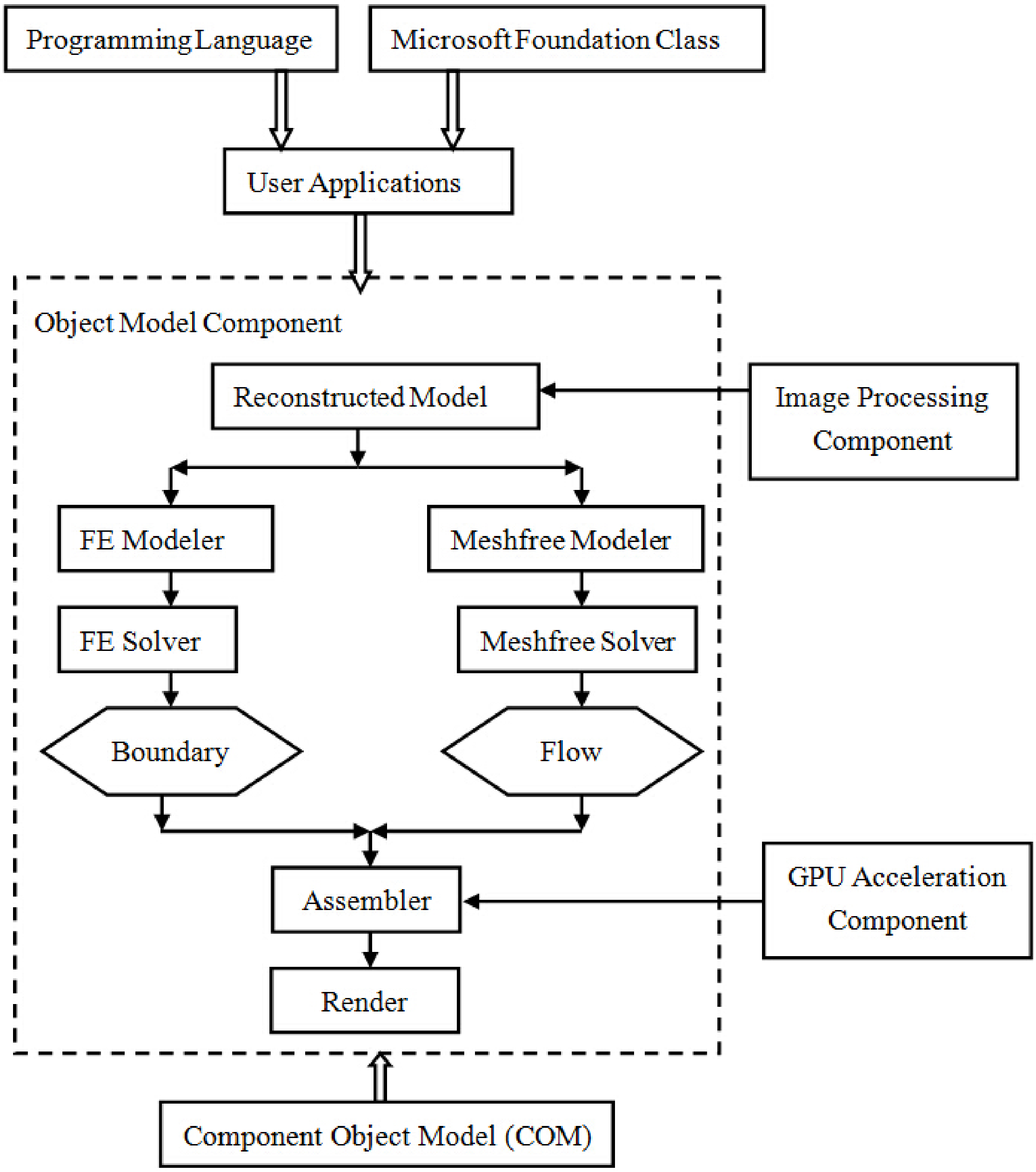


Fig. 3.1 Framework of the proposed component approach: Solid arrows outline the procedural sequence. Non-solid arrows indicate the inheritance. Rectangular boxes indicate the component objects and hexagonal boxes indicate results.

computationally intensive when they are implemented in a CPU-only system. Hence, GPU acceleration component is adopted to optimize the computation in the procedure of searching neighbor particles. The simulation results can be used in graphic rendering of blood flow, drug delivery analysis in vascular system, and investigation of pressure distribution in arterial wall.

3.2 Components and methods

3.2.1 Image processing component

Patient-specific simulation of blood flow requires an accurate reconstruction of patient's vascular model so that the clinicians would be able to analyze the flow rate and trajectory of blood circulation in different applications. In recent years, there is a trend of using general drift flux model to replace patient-specific model [148, 149]. The method assumes a general shape for the blood vessel of patients so that the fluid internal structure can be easily simulated. Their results are probably inaccurate in medical application because the fluid behavior in the blood vessel is highly subject patient-specific, and CFD simulation based on general model cannot be utilized in clinical practice since the geometrical shape of blood vessel is vital for the success of therapy.

The image processing component is used to generate this 3D model from the raw clinical CT images. In this work, a multilayer method is applied to extract the regions of interest before constructing the volumetric model of blood vessel. A 3D thinning algorithm based on Valence Driven Spatial Median (VDSM) is utilized to generate unit-width vessel skeleton. Within the image processing component as shown in Figure 3.2, noise cancellation and 3D assembly are performed to generate a patient-specific vascular model. Details of the reconstruction method and results are introduced in Chapter 4.1.

3.2.2 Meshfree and finite element modelers

The initial motivation for using meshfree method in fluid simulation is to overcome the limitation of grid-based methods in representing the internal structure of liquid [150]. Methods

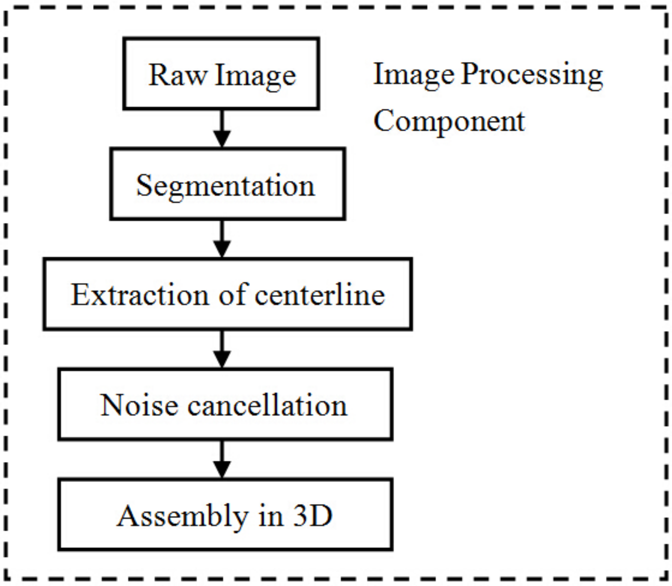


Fig. 3.2 Process flow of the image processing component.

like SPH use a large number of discrete particles to approximate the fluid motion based on input parameters. In a closed or partially-closed system such as blood vessel, the dynamic motion of a simulated particle is only affected by boundary conditions and those surrounding particles within a defined range. The meshfree method performs computation only in the area of interest instead of the whole grid map and so requires less storage and bandwidth. The mass of fluid is also automatically conserved using meshfree method.

Figure 3.3 shows the process flow of the meshfree modeler. Lagrangian particles are first generated in the particle generator, and each individual particle stores its own fluidic parameters and properties. The Navier-Stokes equation which describes the motion of the blood flow is reduced to a simplified form in the model reduction process. A meshfree solver based on an improved SPH method is used to evolve the motion of particle in time. Detailed implementation of the improved SPH method is introduced in Chapter 4.

Force generated due to viscosity and pressure of blood flow cause significant deformation to the vessel wall. A FE-modeler based on lumped element method is used to simulate this soft body deformation phenomenon. The 3D model of arterial wall is modeled as tetrahedral

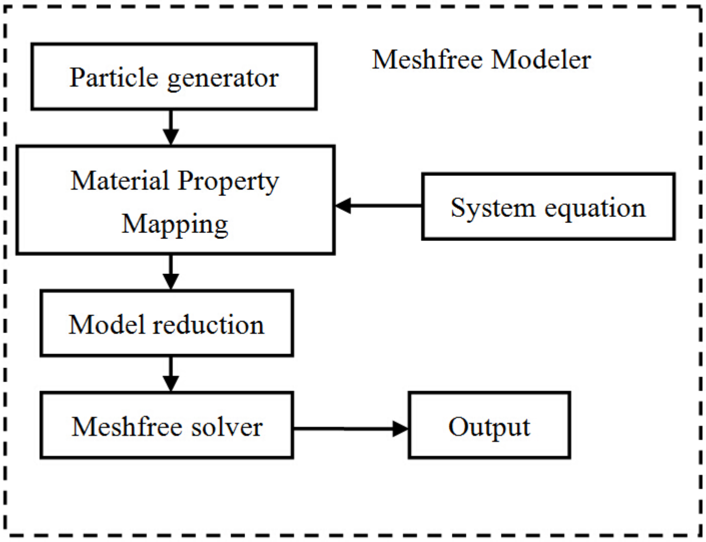


Fig. 3.3 Process flow of the meshfree modeler.

meshes, and the deformation of the tissue is controlled by the damping ratio and stiffness. Lumping the deformation equation on meshed elements allows the simulation to transfer the spatially distributed soft body deformation process into a topological structure consisting of discrete entities.

Physically, such lumping means that the sparsely distributed mass of all elements are reassigned to their vertices in the meshed model. The newly derived mass matrix $\bar{\mathbf{M}}$ consists of a point-like mass at every vertex, with zero mass elsewhere in the mesh segments. The method loses some accuracy in simulation, however, there is always a trade-off between computational speed and realism in real-time simulations. The nonlinear equation of the system is computed using an implicit time integration scheme where the topological backward Euler equation is employed to update the velocities and positions of all vertices. Implementation of this soft body deformation algorithm is shown in Chapter 4.

3.2.3 GPU acceleration component

General purpose GPU (GPGPU) uses multi-tasking computational threads to group cores of GPU in a vector which can be programmed to reduce processing time over large amounts of data. Unlike the mesh-based simulation method which has a straightforward mapping between grid cells and voxels in a 3D texture, meshfree method is not able to be directly parallelized on the GPU due to its irregular structure. Nevertheless, some intensive computations in meshfree method can still be accelerated using GPU such as the searching of neighboring particles [151].

In a computational domain of N particles, SPH reduces the number of particle interactions from N^2 to a very limited number ($< N$) within the smoothing length h. In order to search the neighboring particles, simulation space is subdivided into a grid of uniformly sized cells. Each particle is allocated to the center of a grid cell and the size of the cell is assumed to be the same as the particle for ease of implementation. In static state, a particle covers 8 grid cells in three dimensional space. However, a particle could overlap multiple cells in a dynamic fluid simulation. Therefore, the number of particles in neighboring cells which need to be examined is 27. The grid cells are indexed as a matrix and stored in the shared memory of the GPU, and hence, particles can be allocated into the target grid cells by sorting them based on the grid index. By reading the index of neighbor particles, a neighbor list could be constructed using the neighbor list builder function of CUDA. The size of the neighbor list is determined by the smoothing length h, and the particle interactions outside this range is neglected. Different from the CPU-only system which stores the grid data structure in all time steps, CUDA allows rebuild of the grid data structure from scratch in each time step since the memory used in neighbor search is optimized.

The atomic operation is used to build the grid structure using GPU. In this method, the *updateGrid* CUDA kernel function is used to update the grid structure. One computational thread of GPU is assigned to one particle to find the location of grid cell for the particle,and the *atomicAdd* kernel function is used to atomically increment the cell counter corresponding to the particle locations. The *gridCounter* in global memory basically stores the number of particles in each cell and will be rebuilt in each time step. In this process, the multiple computational thread structure of GPU enables parallel computing of all particles at the same time, and thus, the simulation speed could be improved. Readers who are interested could refer to [152] for detail descriptions of CUDA atomic operation.

3.3 Summary

This chapter presents the proposed component-based modeling framework for blood flow simulation. The framework comprises three independently developed software components, namely image processing component, object model component and GPU acceleration component. The fluid dynamics of blood flow is simulated using a meshfree modeler based on an improved Smoothed Particle Hydrodynamics method, and the interaction with arterial wall is simulated using a Finite Element modeler based on lumped element approach.

Chapter 4

Real-time Simulation of Blood Vessel Deformation

High-fidelity patient-specific model for interactive simulation is important in medical applications. In this chapter, an improved lumped element approach is proposed to simulate deformation of the patient-specific blood vessel in real time by utilizing the GPU for general computing. The blood vessel is segmented and reconstructed from clinical CT-images. The improved lumped element method achieves good simulation realism at high computational speed. Real-time simulation and interactive control are demonstrated in an application on patients with abdominal aorta aneurysm. Realistic simulations of important phenomena such as deformable lumen, self-collision, frictional contact and surface tearing have been achieved.

4.1 Reconstruction of patient-specific vascular system

Three-dimensional reconstruction of patient-specific blood vessel from clinical Computed Tomography (CT) or Magnetic Resonance Imaging (MRI) images is the first step for the simulation of both vessel deformation and vascular drug delivery. Indeed, MRI is more commonly used in today's medical imaging because it provides better contrast between different segments than CT. There is a variety of commercial software which can reconstruct 3D model based on MRI [153, 154]. However, the cost of MRI is relatively expensive and clinical CT imaging devices are still widely used in many developing countries. Therefore, the reconstruction method used in this work is based on clinical CT-images. The method can be easily extended to be MRI-compatible applications since MRI has a clearer gap between the objective organs and the background than CT.

A multi-layer method is applied to extract the raw regions of interest and form the 3D volumetric model of the blood vessel. A 3D thinning algorithm based on Valence Driven Spatial Median (VDSM) is utilized to generate unit-width vessel skeleton. Although the shape of human blood vessel is irregular, it can be modeled as one or more finite beam elements with circular cross section. The 2D cross section of vessel is first segmented from clinical CT-images, and then, Fast Marching Method (FMM) is used to compute and extract the vessel's centerline. The distance transform of the object's boundary is computed by assuming the boundary evolves in normal direction with constant speed. The centerline points are then determined along discontinuities in the distance transform-points where the moving boundary collapses onto itself.

The extracted centerline is usually not smooth and contains a considerable amount of noisy data, which could result in the images being orientated wrongly. Hence, curve fitting is performed to refine the centerline and provide correct orientation for the image slices. The method is based on B-splines which avoid Runge phenomenon of oscillation at the edges of an interval without increasing the degree of the curve function [155]. Along this centerline, each layer of the segmented 2D vessel cross section is assembled into a 3D volumetric model.

The refinement of the 3D model is performed by using a recursive 3D Gaussian Filter which shrinks the mesh surface and eliminates the major noise due to inaccurate CT scanning. By varying the number of recursions, different levels of refinement could be achieved. In order to show the effectiveness of the method, The aorta models of four patients suffering from different severity levels of abdominal aortic aneurysm (AAA) were reconstructed in this work based on their clinical CT-images. Aorta is the main blood vessel that carries blood from the heart to the rest of the body. It runs from the heart, through the chest and abdomen where it branches to supply blood to the legs. Like most blood vessels, aorta is elastic, which allows it to be filled with blood under high pressure. Abdominal aortic aneurysm (AAA) develops when the wall of aorta weakens, causing aortic wall to bulge.

The number of CT-images used in the reconstruction is 55 for each patient. Figure 4.1 shows the scanned CT-images in four different layers. Figure 4.2 demonstrates the resultant 3D models, and Table 4.1 shows the number of faces and vertices in the model of each patient. It can be seen that the reconstructed model generally reflects the severity level of each patient. Patient 1 and 2 are apparently in the early stage of AAA where the normal shape of aorta can still be recognized. On the other hand, AAA is observed to be severely developed in Patient 3 and 4 since the growth of aneurysm significantly deforms the lumen of the aorta.

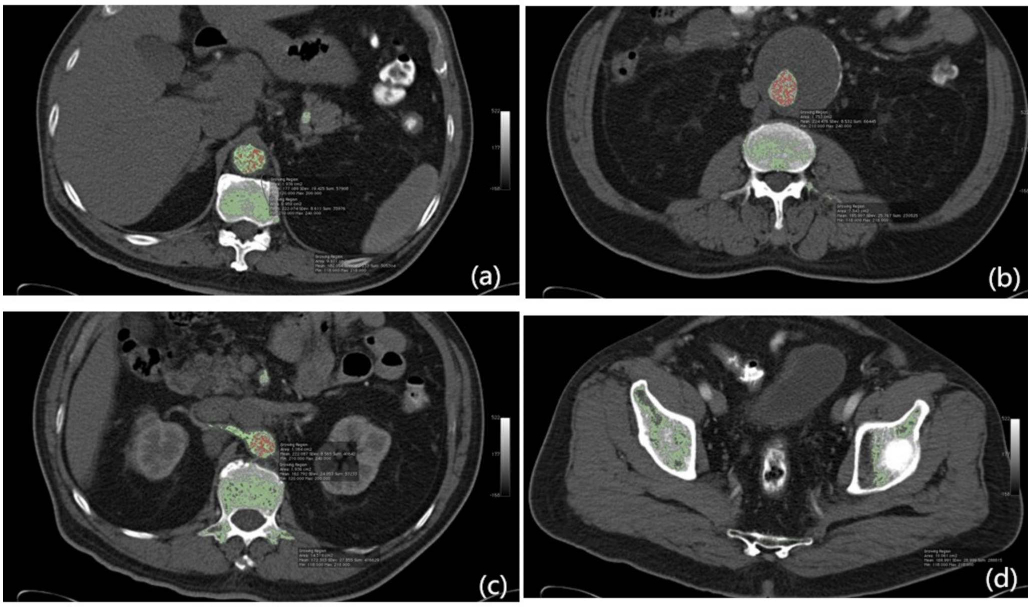


Fig. 4.1 CT-images of aorta in four different layers.

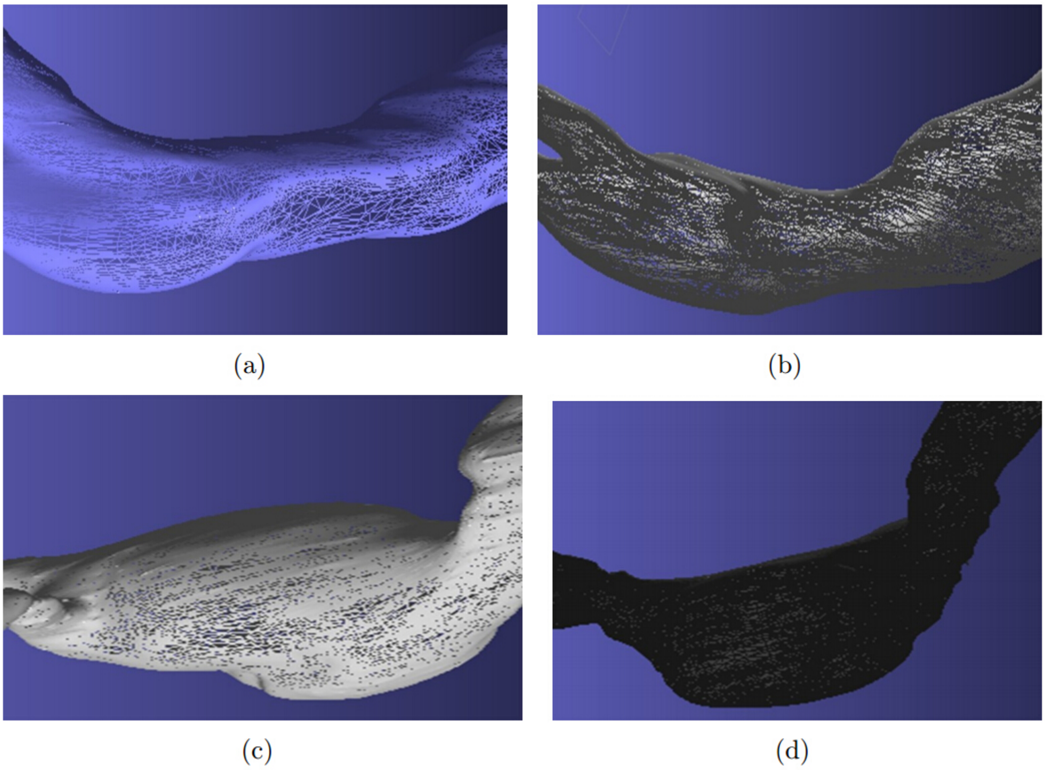


Fig. 4.2 3D patient-specific models of aorta reconstructed from clinical CT-images. (a)(b)(c)(d) corresponds to Patient 1, 2, 3, 4.

Table 4.1 The number of faces and vertices in the reconstructed model of each patient

	Patient 1	Patient 2	Patient 3	Patient 4
Faces	30627	382642	999318	789102
Vertices	15392	191386	500030	394839

In order to simulate vascular blood flow, each segment of the blood vessel is connected to generate the vascular system and the 3D thinning algorithm is used to extract the skeleton of it. This algorithm has the advantages of preserving the model's topological structure while guaranteeing the obtained skeleton will be close to the medial axis of the region [156]. However, the limitation of the original 3D thinning algorithm is that it has problems in generating unit-width curve skeleton, which is important for fluidic simulation. Therefore, a post processing procedure based on valence driven spatial median (VSDM) algorithm is applied to eliminate crowded regions and generate unit-width curves [157].

The post processing begins with valance computation, which calculates the *degree* (number of object voxel in its 26 neighbors) of each voxel on the skeleton. Then, voxels which have degree >2 and at least one of its neighbors has degree >2 are classified as *crowded point*. Next, these crowded points are organized into *crowded regions*, which are sets of 26-connected crowded points. In order to generate a unit-width curve skeleton, all crowded regions must be eliminated. Therefore, a pattern matching is performed to find all object voxels having degree ≤ 2 and 26-adjancent to each crowded region. These voxels are marked with *exit points*. The center of each crowed region can be determined using VDSM algorithm. After that, the shortest path between the center and each exit point is calculated. Any crowded point that is not on any of the shortest path is then removed, and the output skeleton is a unit-width curve as shown in Figure 4.3.

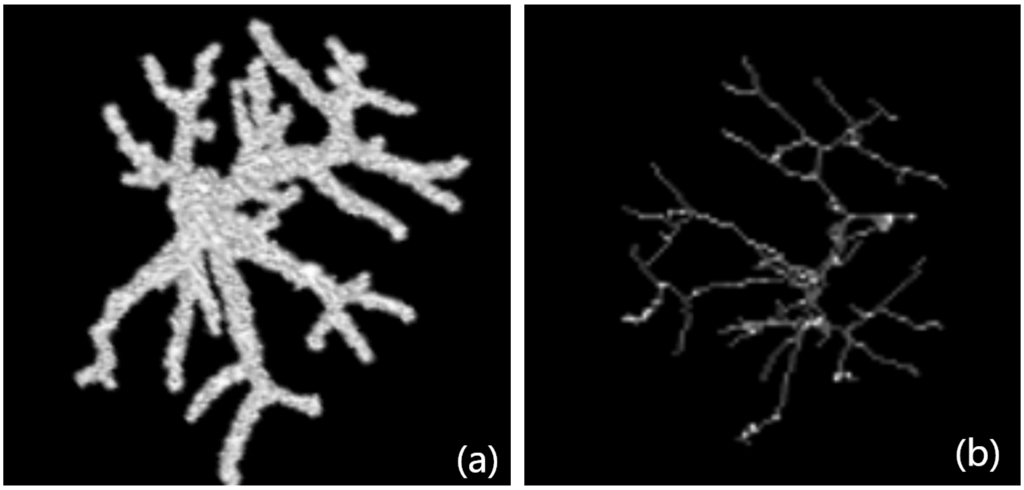


Fig. 4.3 Hepatic vessel reconstruction: (a) 3D vascular system, (b) unit-width skeleton of vessel tree

4.2 Mesh generation

Graphic rendering usually uses a volumetric mesh consisting of 3D polyhedral to represent a simulated deformable object. In mathematics, mesh segment is basically a collection of vertices, edges and faces that defines the shape of the objective polyhedral. The faces of mesh are normally formed by convex polygons such as triangles and quadrilaterals, which could simplify the rendering process.

In the simulation of soft tissue, a particular deformation activity is specified by the displacements of the volumetric mesh vertices. For a meshed model of n vertices, the deformation vector $u \in \mathbb{R}^{3n}$ contains the x, y, z world-coordinate displacements of the model vertices. It can be seen that if the deformation vector u is known, the deformation of an object under external force can be rendered, and collision detection can be performed simultaneously.

In this work, tetrahedral meshes are employed to represent the simulated model in three-dimensional space. Tetrahedron is a polyhedron with 4 faces and 4 vertices. It has been proven in [158, 159] that very fine tetrahedron meshes are sufficient to achieve high solution accuracy. In the field of computer graphics simulation, first order tetrahedron is the most widely used form of mesh technique in representing deformable object. By adding one extra control point in the middle of each tetrahedron, a second-order tetrahedron mesh could be constructed which is able to increase the accuracy of the meshed model during the simulation. Strictly speaking, there is always a trade-off in choosing higher order scheme since it could result in more computations to determine the internal forces and tangent stiffness matrix.

An engineering golden rule is that the cost of computation grows linearly with the order of shape function, while the convergence grows exponentially [160, 161]. However, good convergence does not constantly imply high efficiency in practical applications, especially in interactive simulations. For example, in cases where a volumetric mesh is undergoing frequently contact, there would be a lot of non-smoothness such as derivative discontinuities in the solution. Hence, a higher order mesh to represent smooth deformation fields is not a proper choice for determining the contact threshold in such applications. Reconstruction of patient's blood vessel from clinical CT-images obtains a sufficiently fine model which could be used to describe the deformation activity in the simulation. As long as a proper method of stress and strain rate tensor formulation is employed, realistic rendering of soft tissue could be systematically generated. These will be introduced in the subsequent sections.

4.3 Stress tensor and strain rate tensor formulation

In a deformable object, internal elastic forces are generated in the material to restore the object into its rest shape under external loads. This force is the most vital part in soft tissue simulation since it determines the displacement of all the vertices in the time-varying domain. In order to optimize the computation scheme, it is assumed that the internal forces at location X depend only the deformations in some arbitrarily adjacent elements of X. i.e., the forces at X is not affected by how the material is deforming or previously deformed in the location of some other vertices $Y \neq neighbor\{X\}$. X is the coordinate used to define the undeformed reference geometry. In such implementation, internal force is simulated as a local and topological phenomenon, which can be evaluated based on local quantities such as spatial derivatives of deformation.

In order to define the local amount of deformation, the easiest way is to use the deformation vector u, which is given by

$$u = u\{X\} \in \mathbb{R}^3. \tag{4.1}$$

The deformation vector simply measures the current 3D displacement of X, which is not a good way because its quantity could become too large when a rigid translation is applied. Rigid translation, however, is an arbitrary motion which is independent of deformation.

Therefore, a better measure is to use the gradient of the deformation vector **B**, which is given by,

$$\mathbf{B} = \mathbf{I} + \frac{\partial u}{\partial X} \quad . \tag{4.2}$$

where I is the identity matrix. The deformation gradient is the derivative of each component of the deformation vector u with respect to each component of the reference X, where

$$\frac{\partial u}{\partial X} = \begin{pmatrix} \frac{\partial u_1}{\partial X_1} & \frac{\partial u_1}{\partial X_2} & \frac{\partial u_1}{\partial X_3} \\ \frac{\partial u_2}{\partial X_1} & \frac{\partial u_2}{\partial X_2} & \frac{\partial u_2}{\partial X_3} \\ \frac{\partial u_3}{\partial X_1} & \frac{\partial u_3}{\partial X_2} & \frac{\partial u_3}{\partial X_3} \end{pmatrix} \tag{4.3}$$

Using the gradient of deformation vector could get rid of the unwanted affection from rigid translations, however, it is still not invariant under rigid rotations. That is to say, if a rigid rotation **R** is applied on the object, the measurement of deformation would be $\mathbf{B} = \mathbf{R}$, while there is actually no deformation occurs. Therefore, a more compact solution is to use the strain tensor to measure the deformation activity, which is defined as

$$\mathbf{E} = \frac{1}{2}(\mathbf{B}^T\mathbf{B} - \mathbf{I}), \tag{4.4}$$

where **E** is called the Green-Lagrange strain tensor. In computer graphics, **E** is purely a geometric measure of deformation based on the local deformation field u in the vicinity of X. In surgical simulation, material deformation law is required to be introduced into the model, so that the result can closely resemble the natural human tissue in virtual-reality environment.

The deformation of soft tissue can be viewed as a relationship between the strain rate tensor and stress tensor of an infinitesimal element in the object. Similar to the strain tensor, the stress tensor of a meshed segment is also a diagonal 3×3 matrix. The definition of the stress tensor can be explained as follows: imagine an infinitesimal element of volume dV initially at rest. When deformation happens, the elements in the vincinity are pulling on this infinitesimal element, with some net internal force $f_i dV$. This internal force can be expressed as,

$$\mathbf{f_i} = \text{div}\sigma, \tag{4.5}$$

where σ is called the stress tensor. Both strain and stress tensor have 6 unknown components in the system of equations (not 9, because they are symmetric 3×3 matrix), and hence, they can be related by a 6×6 matrix of coefficients. Nevertheless, this complex mapping method is impossible to be simulated in real-time due to its intensive computations. The solid mechanics community usually solves this problem in an off-line manner computed by supercomputers instead of typical workstations. Therefore, in order to simplify the computations, we model the material as St. Venant Kirchhoff (StVK) model [159], where the stress-strain relationship can be defined as

$$\mathbf{S} = \lambda(tr\mathbf{E})\mathbf{I}_3 + 2\mu\mathbf{E}, \tag{4.6}$$

where S is the second Piola-Kirchhoff stress tensor, $\mathbf{E}$ is the Green-Lagrange strain tensor derived in (4.4). λ and μ are the classic Lame constants, which is given by,

$$\lambda = \frac{\nu E}{(1+\nu)(1-2\nu)}, \tag{4.7}$$

$$\mu = \frac{E}{2(1+\nu)}, \tag{4.8}$$

where E is the Young's modulus, and ν is the Poisson's ratio.

The stress and strain tensor defines the behavior of the object under deformation, which allows the simulation to be performed in a physics-based manner. The derivation of stress and strain tensor needs to be integrated in the continuum equations to render the full features of the deformation phenomenon, which will be introduced in the next section.

4.4 Mathematical modeling of soft tissue deformation

The system equations for the structural deformation of a 3D volumetric object can be derived by the principle of virtual work of Lagrangian mechanics. These equations of motion (called the Euler-Lagrange equation) are a system of second-order ordinary differential equations, which is given by,

$$\frac{d}{dt}(\frac{\partial \mathbf{T}(\dot{u})}{\partial \dot{u}}) + \mathbf{D}(u,\dot{u}) + \mathbf{f_i}(u) + \mathbf{f_{ext}} - \mu\dot{u} = 0, \tag{4.9}$$

where $u \in \mathbb{R}^{3n}$ is the deformation vector of vertices in the model, $\mathbf{f_{ext}} \in \mathbb{R}^{3n}$ is a generalized force arising from external body forces, such as gravity, and $\mu\dot{u}$ is a generalized dissipative force, added to simulate the effect of friction. $\mathbf{T}(\dot{u}) \in \mathbb{R}^{3n}$ is the kinetic energy of the system, $D(u,\dot{u}) \in \mathbb{R}^{3n}$ denotes the damping forces, and $\mathbf{f_i}(u) \in \mathbb{R}^{3n}$ is the internal forces corresponding to the displacement vector u. The last three items in the system equation are not intuitive in mathematics, and the derivations of them need to be further discussed to yield a compact representation for a system of ODEs, which can be solved in generalized coordinates.

The kinetic energy $\mathbf{T}(\dot{u})$ is basically a generalization of the $\frac{1}{2}\mathbf{M}\dot{u}^2$, where

$$\frac{\partial \mathbf{T}(\dot{u})}{\partial \dot{u}} = \mathbf{M}\dot{u}, \tag{4.10}$$

$$\frac{d}{dt}(\frac{\partial T(\dot{u})}{\partial \dot{u}}) = \mathbf{M}\ddot{u}, \tag{4.11}$$

where the matrix $\mathbf{M}$ is called the mass matrix which is composed of all elements in the object. It can be seen that $\mathbf{M}$ is a sparse non-diagonal matrix. If we simply apply this mass matrix in the simulation, it will have many sparse matrix-vector multiplications, which is very computationally expensive. This is the bottleneck for many surgical simulations in simulating complex shape organs such as blood vessel, heart and gallbladder in real time.

Therefore, the lumped element method is used to generate a lumped-mass matrix, which assembles all the row entries onto the diagonal element as it is shown in (4.12),

$$\mathbf{M}=\begin{pmatrix} m_{1,1} & m_{1,2} & \cdots & m_{1,j} \\ m_{2,1} & m_{2,2} & \cdots & m_{2,j} \\ \vdots & \vdots & \ddots & \vdots \\ m_{i,1} & m_{i,2} & \cdots & m_{i,j} \end{pmatrix} \to \bar{\mathbf{M}}=\begin{pmatrix} \sum_{b=i}^{j} m_{1,b} & 0 & \cdots & 0 \\ 0 & \sum_{b=i}^{j} m_{2,b} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sum_{b=i}^{j} m_{i,b} \end{pmatrix} . \tag{4.12}$$

Physically, such lumping means that the sparsely distributed mass of all elements are reassigned to their vertices in the meshed model. The newly derived mass matrix $\bar{\mathbf{M}}$ consists of a point-like mass at every vertex, with zero mass elsewhere of the mesh segments. It is obvious that such lumping would lose some accuracy in simulation. However, the reconstructed model used in the simulation is considered as fine mesh and could compensate for the loss of accuracy in the final rendering. By using the lumped element approach, the computational cost would be largely reduced.

In order to simplify the computation, the damping item $\mathbf{D}(u,\dot{u})$ is modeled as memoryless, which means it is independent of time in the simulation. There is no general method for the modeling of damping ratio in human tissue because it is rather non-homogenous in nature [162]. In this work, a local Rayleigh damping model is employed which is commonly used as a viscous damper for soft tissue simulation [163, 164]. The mathematical description of the model is given by,

$$\mathbf{D}(u,\dot{u})=\left(\alpha\bar{\mathbf{M}}+\beta\mathbf{K}(u)\right)\dot{u} \quad , \tag{4.13}$$

where α and β are two positive real-valued parameters used to control the damping model. Specificly, α has the effect of low frequency damping and β has the effect of high frequency damping. The model can be further simplified into linear Rayleigh damping model, where $\mathbf{K}(u)$ is replaced by an averaged constant K. In this work, $\alpha = 0$, and $\beta = 0.22$ are used, which make the system a little overdamped.

The internal force $\mathbf{f_i}(u)$ is the most difficult part in the simulation. Some literatures in solid mechanics use the derivative of strain energy to compute the internal forces, which is not efficient in real-time simulation since the cost of computation is too expensive. In this simulation, a co-rotational finite element method based on the work in [165] is used to simulate the deformation of soft tissue due to internal forces.

If the internal force is directly modeled in a vertex-based geometry, the simulation would have to compute a large sparse system matrix at every time step. Instead, the local stiffness matrix $\mathbf{K_e}$ for each element e is computed, and the final stiffness matrix $\mathbf{K}$ of the entire model is assembled by the individual $\mathbf{K_e}$ of all tetrahedron elements.

In an element e, a, b, c, and d are used to denote four vertices of the tetrahedron. $u_e \in \mathbb{R}^{12\times 1}$ is the displacement vector of the four vertices. The strain rate tensor $\xi_\mathbf{e} \in \mathbb{R}^{6\times 1}$ within an element is given by,

$$\xi_\mathbf{e} = \mathbf{G_e} u_e \quad , \tag{4.14}$$

where $\mathbf{G_e}$ is a 6×12 matrix which maps the displacement vector u_e to the strain tensor $\xi_{\mathbf{e}}$. In a tetrahedron geometry, $\mathbf{G_e}$ has been derived in [60], which is given by,

$$\mathbf{G_e} = \begin{pmatrix} -p_x(bcd) & 0 & 0 & -p_y(bcd) & 0 & -p_z(bcd) \\ 0 & -p_y(bcd) & 0 & -p_x(bcd) & -p_z(bcd) & 0 \\ 0 & 0 & -p_z(bcd) & 0 & -p_y(bcd) & -p_x(bcd) \\ p_x(cda) & 0 & 0 & p_y(cda) & 0 & p_z(cda) \\ 0 & p_y(cda) & 0 & p_x(cda) & p_z(cda) & 0 \\ 0 & 0 & p_z(cda) & 0 & p_y(cda) & p_x(cda) \\ -p_x(dab) & 0 & 0 & -p_y(dab) & 0 & -p_z(dab) \\ 0 & -p_y(dab) & 0 & -p_x(dab) & -p_z(dab) & 0 \\ 0 & 0 & -p_z(dab) & 0 & -p_y(dab) & -p_x(dab) \\ p_x(abc) & 0 & 0 & p_y(abc) & 0 & p_z(abc) \\ 0 & p_y(abc) & 0 & p_x(abc) & p_z(abc) & 0 \\ 0 & 0 & p_z(abc) & 0 & p_y(abc) & p_x(abc) \end{pmatrix}^T , \tag{4.15}$$

$$p(uvw) = u \times v + v \times w + w \times u, \tag{4.16}$$

where p_x, p_y, p_z are the respective first, second and third component of p. Considering the stress-strain relationship, the stress tensor σ_e within the element is given by,

$$\sigma_{\mathbf{e}} = \mathbf{H_e} \xi_{\mathbf{e}} \tag{4.17}$$

where $\mathbf{H_e}$ is a 6×6 matrix characteristic of the element material based on Poisson's ratio ν and Young's modulus E. In the tetrahedron mesh structure, $\mathbf{H_e}$ is given by,

$$\mathbf{H_e} = \begin{pmatrix} \gamma+\chi & \gamma & \gamma & 0 & 0 & 0 \\ \gamma & \gamma+\chi & \gamma & 0 & 0 & 0 \\ \gamma & \gamma & \gamma+\chi & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\chi}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\chi}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{\chi}{2} \end{pmatrix}, \tag{4.18}$$

where

$$\gamma = \frac{E\nu}{36V_e(1+\nu)(1-2\nu)} \quad , \tag{4.19}$$

$$\chi = \frac{E}{36V_e(1+\nu)} \quad , \tag{4.20}$$

and V_e is the volume of the tetrahedron element. Therefore, the internal force of the element f_{ie} can be obtained by,

$$\mathbf{f_{ie}} = \mathbf{K_e} u_e, \qquad \mathbf{K_e} = \mathbf{G_e}^T \mathbf{H_e} \mathbf{G_e}. \tag{4.21}$$

However, the internal force derived in this way still has a limitation. If the element is under a rigid rotation, the deformation vector would be changed while actually no deformation activity has occurred. Therefore, a local rotational matrix $\mathbf{R_e}$ is added to compensate for this rigid rotation. In each tetrahedron, the local frame is represented by a rigid rotation matrix $\mathbf{R_e}$, and its transpose is used to align the deformed tetrahedron with its reference, undeformed shape. The displacements and forces are computed in the rotated coordinate system, and then, they will be transformed back to world coordinates. Thus, the final form of the element's internal force is given by,

$$\mathbf{f_{ie}} = \mathbf{R_e}\mathbf{K_e}\mathbf{R_e}^T u_e \quad . \tag{4.22}$$

In a meshed model consists of n elements, the total internal force $\mathbf{f_i}$ is an assembly of all elements. In the literatures, this framework is able to simulate deformable object of moderate complexities, such as a mesh model consisting of hundreds of vertices, in a CPU-only system. However, as it has been shown in section 4.1, a fine mesh of patient-specific blood vessel reconstructed from clinical CT-images is highly complex and irregular. Assembly of the internal forces in such kinds of complicated mesh model is difficult for the capability of current CPU technology, which executes a computational task in series. Hence, GPU parallelization is used to accelerate the simulation so that it can be performed in real time. Implementation of GPU parallel computing is introduced in the next section.

4.5 GPU parallelization

Due to the advancement of GPU technology in the past ten years, a number of algorithms have been developed to increase the efficiency of simulation through parallel computing method. The motivation of research in this field is because the new generation multi-core General Purpose GPU (GPGPU) has hundreds of cores available for texture and volume rendering. It has been reported that GPU computing is able to provide at least 10 times acceleration for graphics applications compared with CPU-only systems [127]. NVidia has developed a variety of embedded algorithms of parallel computing based on CUDA (Compute Unified Device Architecture) platform, and some of them have already been used in surgical simulations [30, 31, 60, 128, 133].

In this work, the method of assembly by nonzero entries using global memory is employed for the task parallelization of the simulation [166]. Specifically, one computational thread is used to compute one tetrahedron and the result is stored independently in the corresponding vector matrix, and the forces on each vertex are accumulated using an array that stores each vertex's indices of all connected tetrahedrons.

As shown in Figure 4.4, the element data is written in coalesced memory transactions to global memory for later reduction into a system of equations. Since the reduction process operates on element data which are stored in global memory and there are no global synchronization primitives, the computation and assembly of the element data must be performed using separate kernels. To simplify the computations, the program makes use of the function that elements can be grouped in the transaction, and the total number of transactions with global memory can be reduced by prefetching all the meshed data that a thread block will require and sharing it between the set of elements to be computed [166]. The contiguous entries of the matrix value array are then assembled in shared memory which coalesced writes are performed into matrix's value array when all nonzero entries have been computed. By using GPU parallelization for general computing, it is possible to perform real-time simulation of blood vessel deformation on a typical workstation.

4.6 Simulation of soft tissue division

Realistic simulation of soft tissue division problem is important in surgical application. Based on the biomechanical properties, human tissue is only able to sustain certain range of load before tearing is happened. As it has been discussed in the previous sections, deformation of soft tissue could be simulated using lumped element method. However, effective simulation of soft body division is limited in the literature. In fact, soft tissue division is an inevitable surgical procedure in many types of vascular surgery including Endoscopic Vessel Harvesting (EVH), laparoscopic bypass surgery, and scaffold deployment in aorta. When a virtual surgery

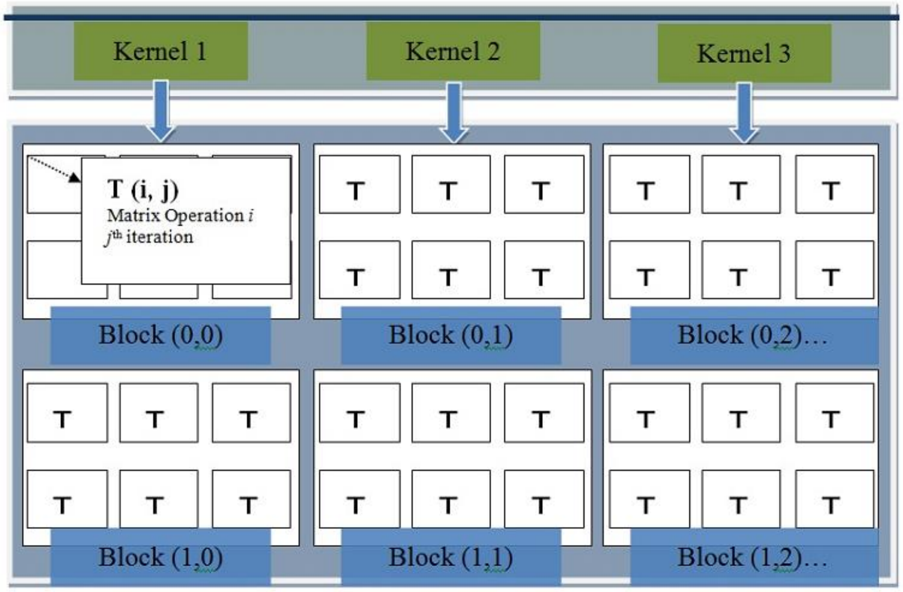


Fig. 4.4 Architecture of GPU parallelization in matrix assembly.

is carried out, it is desirable for the simulation to render the effect of soft tissue division in various types of environment, so that the surgical training and planning could be evaluated based on the simulation result.

In this work, the division of soft tissue is simulated by using a tear factor based on the biomechanical property of the respective tissue. The magnitude of the tear factor is utilized to define the force limit on the connections between each meshed segment based on the biomechanical properties of the simulated soft tissue. This force limit is modeled as the threshold kernel function to control the initial condition of soft tissue division. Once the internal force exceed the limit, the meshed segment will be dethatched from the model. Figure 4.5 shows the process flow of the proposed method.

Allowing the meshed model to be divided is only the first step. The most difficult part is to define another threshold to control the end condition of the soft tissue division procedure. Different from the conventional way of using the geometry of vertices in the system as the threshold, a novel idea of using non-linear physical process is employed to describe this

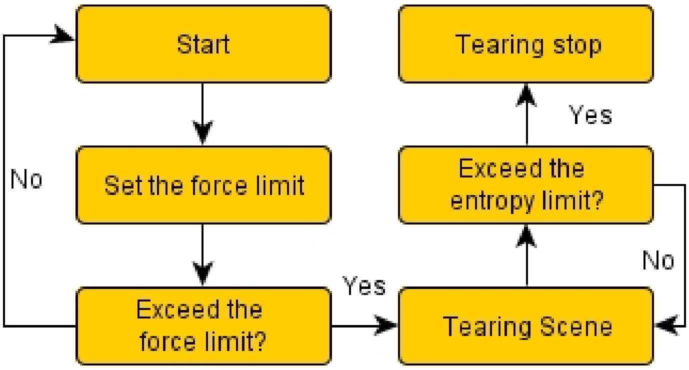


Fig. 4.5 Flow chart of soft tissue division simulation.

phenomenon in mathematics. Specifically, the entropy of the model during runtime is used as the threshold, which is given by

$$S = c\sum_{i} p_i(t)\ln p_i(t) \tag{4.23}$$

where S is the entropy value of the system, c is the Boltzmann constant and p_i is the probability that the system is in the i-th microstate.

The undeformed soft body preserves the lowest entropy value as all the vertices are at the initial steady state. When some elements are detached in a tearing motion, the meshed model would become a disordered system due to its highly irregular shape under twisting, elongation and contraction. Such change would dramatically increase the randomness of the system which would then increase the entropy of the system. When the threshold in the corresponding buffer cannot hold the entropy generated by the new vertices anymore, the simulation stops.

The concept of using entropy as the threshold closely resembled the biological mechanism of tissue division, which is basically an energy releasing process. On the other hand, the conventional way only considers geometrical deformations, which is essentially a ‘solid’ threshold with no regard to the law of physics. Entropy threshold claims more flexibility to simulate the division of soft tissue, and it also describes structure randomness and energy state of the model.

4.7 Results and discussion

A NVDIA Quadro FX 4600 graphics card in an Intel Xeon CPU @3.00GHz was used as the computing platform for the simulation. In order to evaluate the performance of the proposed method in soft tissue simulation, an experiment of applying external virtual force on the reconstructed blood vessel model of patient 1 was implemented. As it is shown in Table 4.1, the original reconstructed models are too complicated to be used in a real-time simulation, and hence, the models were compressed before the mesh generation. The meshed results of the simulated models are shown in Figure 4.6. Compare with Figure 4.2, the compressed models still yield a fine shape convergence with respect to the original reconstructed models. The severity level of AAA in each patient can be easily identified based the meshed model through visual inspection, which is important in surgical application since it allows clinicians to plan and evaluate surgical procedures in a virtual-reality environment. The biomechanical parameters of patient 1 using in the simulation is shown in Table 4.2

Table 4.2 Simulation parameters of Patient 1's blood vessel deformation

Parameter	
Young's Modulus	4.23 MPa
Poission's ratio	0.23 MPa
Number of vertices	1427

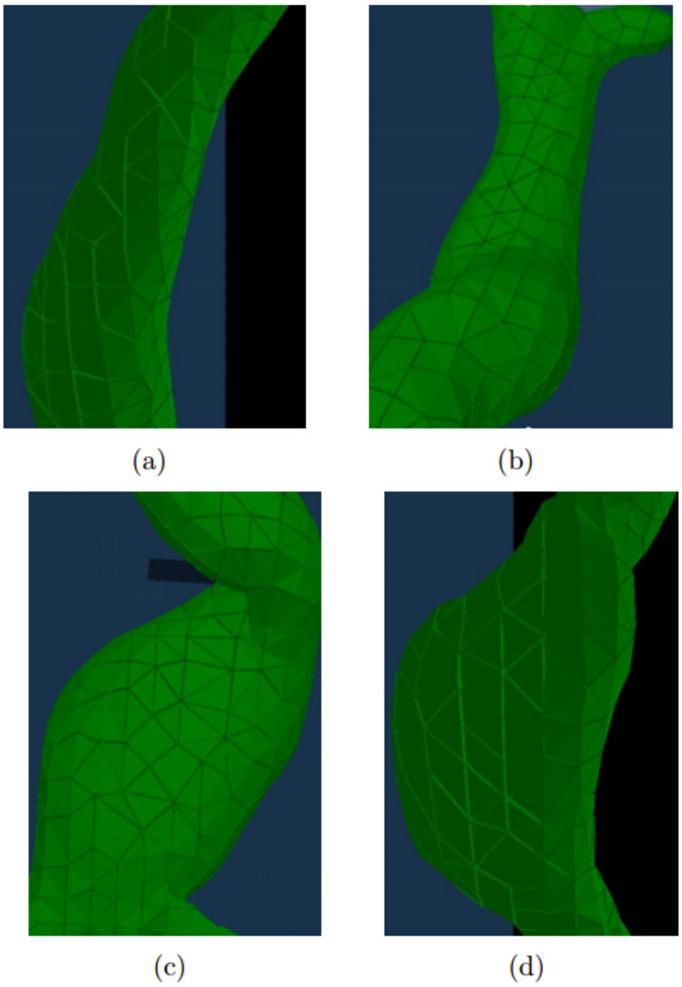


Fig. 4.6 Meshed models of the four patients after compression (In the order of (a),(b),(c), and (d)) .

4.7.1 GPU acceleration

At first, two identical meshed models were used to compare the computational speed between the CPU-only system and GPU-accelerated system in real-time simulation. The CPU-only system was only able to handle 8 contact points while maintaining the simulation to be performed in real time. On the other hand, the maximum number of contact points that can be computed is 128 which consist of 768 constraints by using GPU acceleration. In medical simulations, the ability to compute more contacts points in the deformation activity could allow the program to include more degree of freedoms in the design of virtual surgical experiment.

In order to quantitatively evaluate the acceleration of the assembly by nonzero entries using global memory method through GPU parallel computing, simulations of the four patients' blood vessel deformation with the same boundary pressure were implemented. Constant 5 MPa boundary pressure was applied at the middle of the patients' blood vessel models, and 80 contact points were evaluated in the simulation. The comparison in computational speed is shown in Table 4.3. In this scenario, the simulation based on CPU-only system was not able to be performed in real time, and the rendering speed was shown to be between 1.5 to 2.1 Frame Per Second (FPS). By employing GPU acceleration in the simulation, the program was able to render the graphical results of blood vessel deformation at the rate of 21.2 to 29.8 FPS. It can be seen that the simulation using GPU yields a considerable amount of speed-up compare with CPU-only system. Based on the suggestions from the clinician in National University Hospital (Singapore), the value of interactive surgical simulation can be reflected in practical applications only if the simulation is able to be performed in real time. Hence, the acceleration of the program by using GPU parallel computing is important for the technical success of the work.

Table 4.3 Comparison of computational speed for CPU-only and GPU-accelerated simulations

		Rendering speed (FPS)	
Subject	Number of vertices	CPU	GPU
Patient 1	1427	2.1	29.8
Patient 2	1588	2.0	28.5
Patient 3	1953	1.7	23.3
Patient 4	2251	1.5	21.2

4.7.2 Aortic deformation and rupture

The reconstructed model of patient 1 was used to demonstrate the simulation of aortic deformation and rupture. The number of vertices of the meshed model is 1427. Interactive simulation of the meshed model was also tested by using a computer mouse as the force rendering device. A cursor controlled by the computer was shown on the screen when simulation started. The computer mouse could be replaced by a force feedback mouse to be used in the simulation with haptic rendering.

According to [167], an aorta with AAA would be torn under a load of 10-12 MPa. In this experiment, a boundary pressure varies from 8 to 12 MPa was continuously applied at the middle of the patient's aorta, and solid boundary condition was applied at the two ends of the blood vessel to allow the deformation to happen. In order to ensure consistent measurement, a pre-defined trajectory was used to progressively remove a section of aorta when excessive load was applied. It is observed that the simulation is performed under realistic and stable interactions, and the simulated aorta model of patient 1 was torn when the load was increased to 10.4MPa. Figure 4.7 demonstrates the whole process of aortic deformation and rupture in the simulation, and Figure 4.8 shows the moment when a part of the meshed model was detached from the aorta. Aortic rupture is an important symptom of AAA which could cause severe pain in the lower back, flank, abdomen or groin. The mortality of aortic rupture due to AAA is as high as 90 percent. It has been shown that 65 to 75 percent of patients die before they arrive at the hospital and up to 90 percent die before they reach the operating room [46]. Evaluation of the aortic deformation and rupture using the proposed simulation could reduce the surgical risk and improve the effectiveness of surgical training.

4.7.3 Simulation of 'cloth-type' soft body deformation

The soft body simulation method used in this work can be extended to simulate other types of soft-bodied objects. Figure 4.9 shows a simulation of three different stiffness

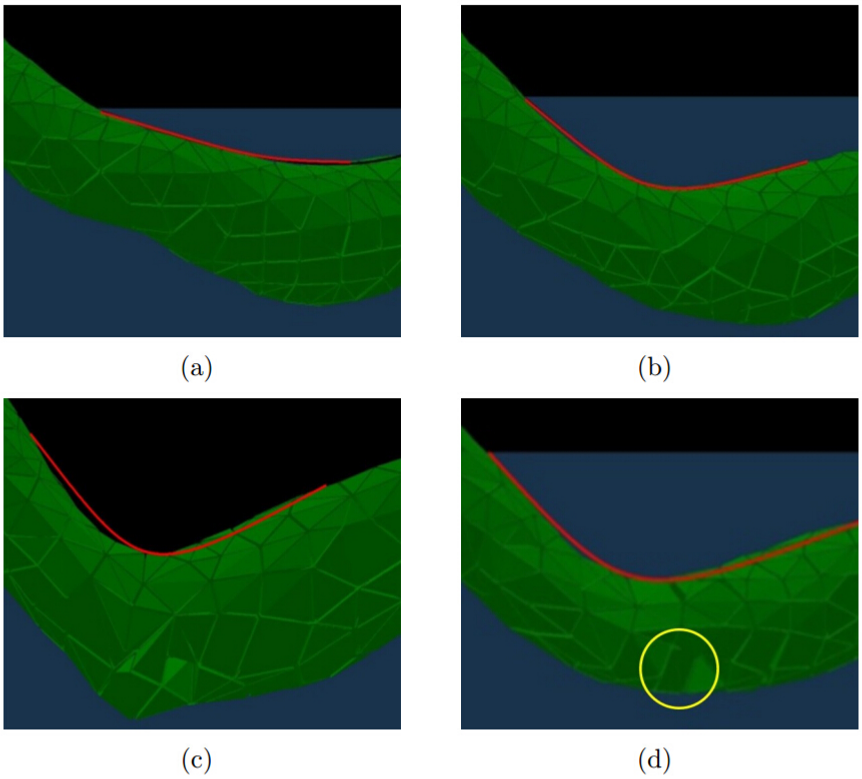


Fig. 4.7 Simulation of aortic deformation and rupture: a) original aorta model, b) deformation under external force, c) the aorta is about to be torn, d) the final torn aorta model (The red line indicates the curvature of deformation and the yellow circle in (d) highlights the part where fracture occurred).

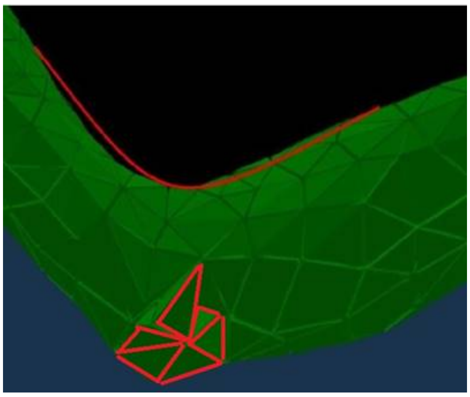


Fig. 4.8 Simulation of aortic rapture when a part of the meshed model was detached from the aorta

'cloth-type' soft bodies in real time. The material properties are shown in Table 4.4. Same magnitude of external force was applied on the three objects to create the deformation activity.

Compare with the results of the early work in Section 2.2.2, it can be seen that the realism of tissue deformation has been enhanced based on visual inspection. The deformation process appeared to be more fluent and the different behaviors for the three stiffness properties were clearly observed in the simulation.

Table 4.4 Material properties for the simulation of 'cloth-type' soft body deformation

	Material 1	Material 2	Material 3
Stiffness	5 MPa	20 MPa	100 MPa
Thickness	0.5mm	0.5mm	0.5mm

The reconstruction of blood vessel models from clinical CT-images realistically reflects the severity of symptoms for the four patients with AAA. The general shape of the meshed model converges largely with the original reconstructed model, and hence, it can be used in the real-time simulation of blood vessel deformation. Reviewing the conventional methodologies, the bottleneck of simulation appears to be the sparse matrix-vector multiplication. The proposed lumped element method largely reduces this costly computation by using lumped mass matrix to perform the simulation. Parallel computing performed on GPU prevents costly CPU-GPU transfers of data and provides a significant speed-up. This speed-up is important because it allows large and precise meshed model to be used in real-time simulation, which is also critical for the technical success of surgical simulation. Besides, the simulation of aortic rupture using entropy threshold is an exploratory method to set the end condition of soft body division process. The results are shown to be stable with no numerical overflow, and a segment of the meshed elements were successfully detached from the aorta model in the simulation. Parallel computing performed on GPU prevents costly CPU-GPU transfers of data and provides a significant speed-up. This speed-up is important because it allows large and precise meshed model to be used in real-time simulation, which is also critical for the accuracy and realism of surgical simulation.

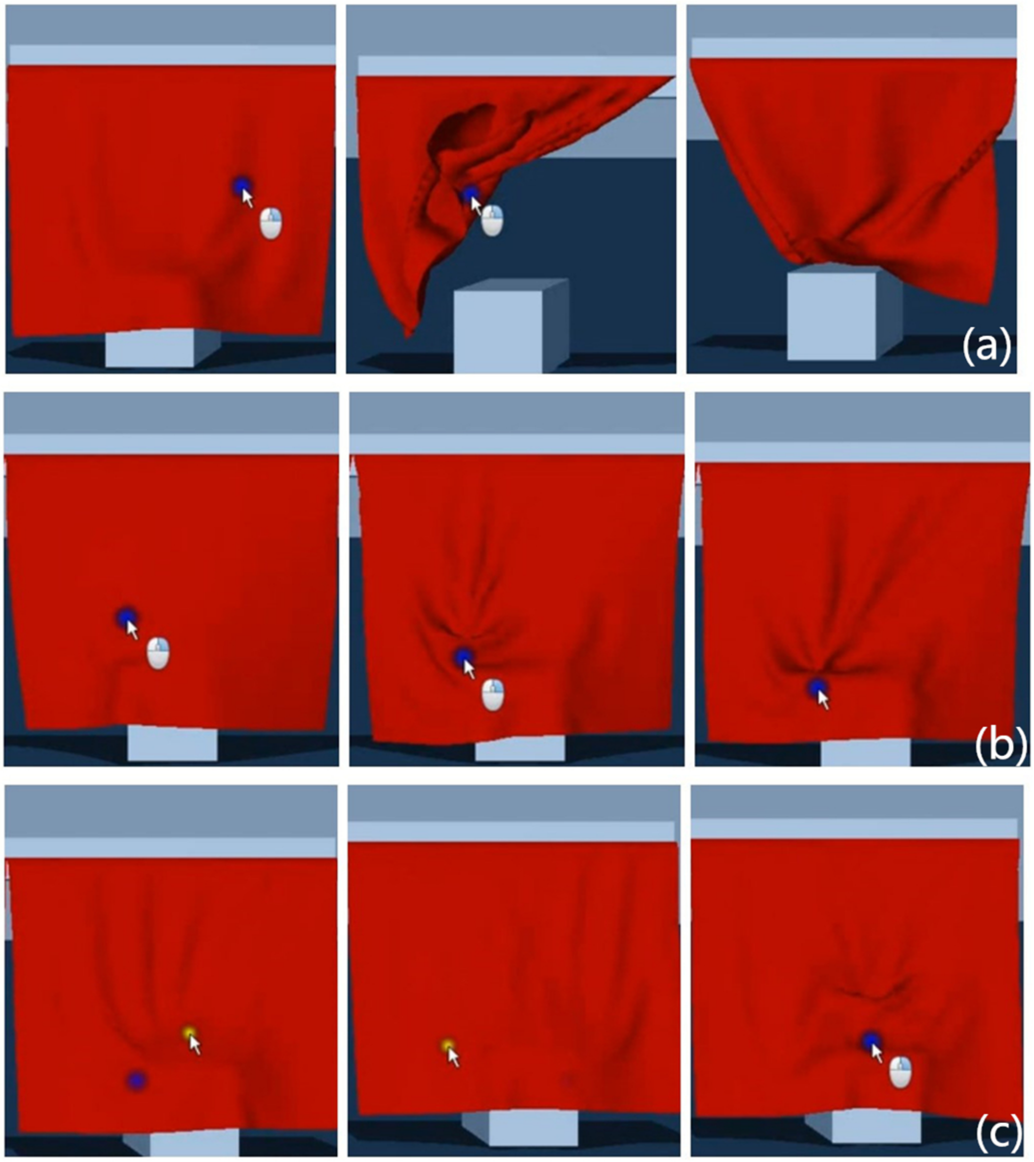


Fig. 4.9 Simulation of ‘cloth-type’ soft body deformation with different stiffness.

There is always a trade-off between realism and computational complexities in real-time simulation. In this work, the simulation was designed to be interactive while maintaining the basic law of physics in the deformation activity. The work still needs to be further validated by biological experiments so that it can be used in medical applications. However, such experiments are generally costly and difficult to be implemented on animal or human subjects.

4.8 Summary

The chapter presents the patient-specific simulation of blood vessel deformation. The model was first reconstructed from clinical CT-images using multi-layer method and meshed using tetrahedron elements. The deformation of blood vessel was simulated using lumped element method, and the computational process was paralleled using the method of assembly by nonzero entries using global memory based on the multi-core architecture of GPU. Parallel computing performed on GPU prevents costly CPU-GPU transfers of data and yielded a significant speed-up. The local deformation of each meshed segment can be computed independently by the multi-core GPU to reduce the computationally cost. The soft body division process was simulated using entropy threshold, and the simulation of aortic deformation and rupture was demonstrated on a patient with AAA. The proposed method can be validated and further refined by using biological experiments so that it can be used in practical medical applications.

Chapter 5

Hemodynamic Simulation of Blood Circulation

Study of hemodynamics is normally based on existing cases of patients with specific anomalies. Evaluations and investigations of new cases are limited in practice due to the need for accurate and timely prediction of blood flow on patient-specific models. Simulations using reconstructed models of patient-specific vascular system have been reported in [110–112]. However, fluid dynamics of blood flow is highly complex. To our knowledge, realistic and fast simulation of blood flow is lacking in medical simulation systems. Bio-fluid simulation should be sufficiently accurate and efficient to represent the underlying physical parameters, and it should also be able to simulate viscous and inviscid flows for multi-phase fluids and mass transfer during steady state inhalation. Generally, such simulation should be defined in a way that, given specific input parameters and scene geometry, a fluid simulator evolves the motion of the fluid forward in time based on Navier-Stokes equations.

In this chapter, two novel approaches are proposed to simulate blood flow of patients. The first approach is based on an improved Smoothed Particle Hydrodynamics (SPH) method, which is dedicated to rendering a 3D graphical simulation. The second approach is based on an improved Finite Particle Method (FPM), and the method was implemented in two cases of blood flow simulations with respect to hand and heart circulation with blockages. In this dissertation, physics-based simulations of blood flow have been performed on patient-specific models using the proposed methods. The results demonstrated that meshfree method yields good simulation realism at high computational speed.

5.1 Modified SPH method

The initial motivation of using meshfree method in fluid simulation is to overcome the limitations of grid-based methods in representing internal structure of liquid. Methods like SPH use a large number of discrete particles to approximate the fluid motion based on the input parameters. In the conventional particle-based method, the motion of a particle is independently evaluated based on the force interactions with all the other particles in the problem domain. In a system of N particles, the number of computations would be N^2 in every time step, which is generally regarded as intensive in practical applications. In order to reduce the computational cost, SPH method was proposed, where the dynamic motion of a simulated particle is only affected by boundary conditions and those surrounding particles within a defined range.

In a closed or partially-closed system like a blood vessel, the magnitude and orientation of external force vectors can be easily customized using SPH method based on the problem's requirements. Hence, different hemodynamic behaviors can be modeled by simply changing the pre-defined settings of the problem domain. Another advantage of using meshfree method in hemodynamic simulation is that the mass and momentum are automatically conserved. Therefore, the numerical dissipation, which is common in grid-based methods, could be eliminated in the time-evolving simulation of blood flow.

In SPH method, a smoothing kernel is used to define a scalar weighting function in the vicinity of x_i of particle i via $W(|x-x_i|)$. In each simulated particle of the problem domain, the kernel determines which group of surrounding particles would contribute to the force interactions that affect the particle's motion in a single time step. In order to maintain the convergence of the simulation, the smoothing kernel function needs to be normalized, which means $\int W(|x-x_i|)dx = 1$, so that the conservation of total mass could be achieved. In this work, a quartic kernel function is proposed which is given by,

$$W(|x-x_i|,h) = \frac{315}{208\pi h^3}\begin{cases} \frac{2}{3}-\frac{8}{9}(\frac{|x-x_i|}{h})^2+\frac{19}{24}(\frac{|x-x_i|}{h})^3-\frac{5}{32}(\frac{|x-x_i|}{h})^4 & \text{if } 0 \le |x-x_i|/h \le 2 \\ 0 & \text{if } |x-x_i|/h > 2. \end{cases} \tag{5.1}$$

where h is the defined smoothing length. Only those particles within this length would be included in the computation of force interactions with the target particle. Other particles in the problem domain would be 'smoothed' by the proposed kernel function.

It can be seen that the kernel is symmetrical around the simulated particle since the result only depends on $|x-x_i|$, which is the absolute value of the distance to the target particle. Comparing with that of the conventional cubic splines where,

$$W(|x-x_i|,h) = \frac{315}{208\pi h^3}\begin{cases} \frac{2}{3}-(|x-x_i|/h)^2+\frac{1}{2}(|x-x_i|/h)^3 & \text{if } 0 \le |x-x_i|/h \le 1 \\ \frac{1}{6}(2-|x-x_i|/h)^3 & \text{if } 1 \le |x-x_i|/h \le 2 \\ 0 & \text{if } |x-x_i|/h > 2, \end{cases} \tag{5.2}$$

their function values, first and second derivatives are shown in Figure 5.1. The function value of the proposed quartic kernel is very close to the cubic splines with the same peak value of $\frac{2}{3}$, and monotonically decreases with the increase in the distance. On the other hand, the quartic kernel yields a smoother second derivative compare with that from cubic spline, and thus, it could reduce the tensile instability caused by meshfree method and produce more stable SPH formulations.

The tensile instability could be explained in an example of solid mechanics simulated using SPH method. As it has been illustrated previously, the force interactions between these particles are computed from the equations of motion formulated by the SPH particles. The particles would repel each other when the force is positive, and attract each other when the force is negative. However, the attraction can lead to an instability which would result as particles forming small clumps when a gaseous or fluidic subject is simulated [168]. This instability was first studied in detail by Swegle et al. who related the instability to a combination of the negative pressure and the sign of second derivatives of the SPH interpolating kernel [169]. It has been shown in [170] that the stability properties of SPH depend strongly on the smoothness of the second derivatives of kernel function. Readers who are interested could refer to [171, 172] which illustrated in depth the instability in fluidic simulations.

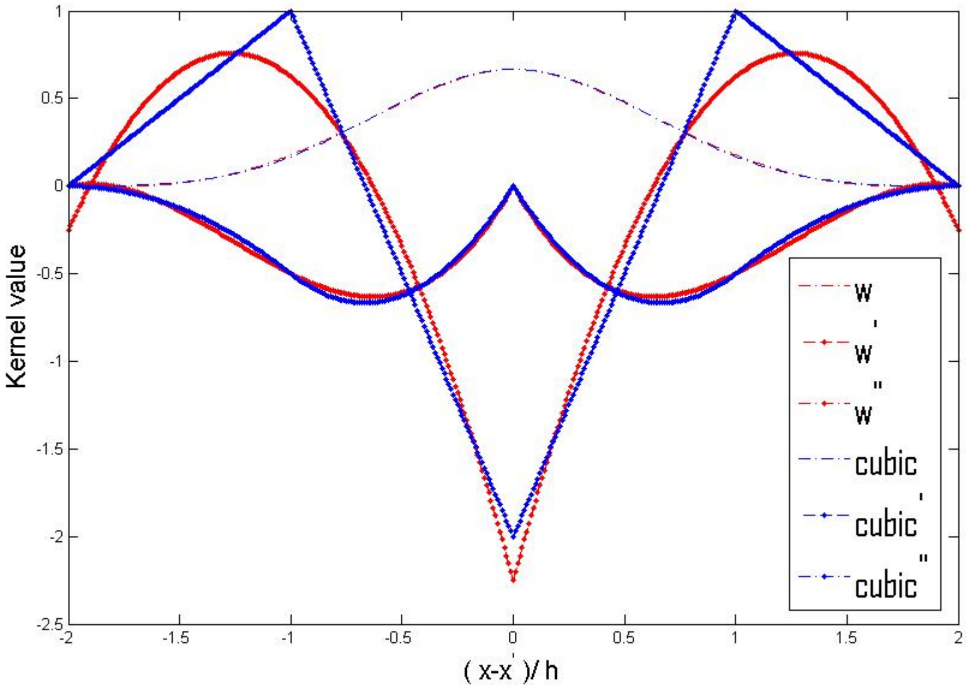


Fig. 5.1 Comparison of the quartic kernel function and cubic splines with respect to their function value and first two derivatives.

After eliminating the tensile instability, the kernel function is used to construct the smooth density field, which is given by,

$$\rho(x) = \sum_j m_j W(|x - x_i|) \quad , \tag{5.3}$$

where m_j is the mass of particle j. The value of the mass is subjected to the type of simulated fluid. Different mass profiles could be assigned in a unified problem domain to generate the effect of multi-phase fluid, which is important for the simulation of blood circulation since the gas-phase in human blood could affect the flow trajectory in the vascular system. As the smooth kernel has already been normalized, the total mass of the fluid can be computed by taking the integral of density field,

$$\int \rho(x)dx = \sum_j (m_j \int W(|x - x_i|)dx) = \sum_j m_j \quad . \tag{5.4}$$

By rearranging (5.3), a smooth field A_s of fluidic parameter A and its first derivative ∇A at any point x in the problem space is given by,

$$A_s(x) = \sum_j m_j \frac{A_j}{\rho_j} W(|x - x_i|), \tag{5.5}$$

$$\nabla A_s(x) = \sum_j m_j \frac{A_j}{\rho_j} \nabla W(|x - x_i|), \tag{5.6}$$

The smooth field is then used in the Navier-Stokes (N-S) equations to represent the corresponding fluidic parameters. The conservation of mass and conservation of momentum in N-S equations are given by,

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho v) = 0 \quad , \tag{5.7}$$

$$\rho(\frac{\partial v}{\partial t} + v \cdot \nabla v) = f_{ext} - \nabla p + \mu \nabla^2 v \quad , \tag{5.8}$$

where v is the velocity field, ρ is the density field, f_{ext} is the external body force, μ is the viscosity of the fluid.

It can be seen that the SPH method has several benefits in solving the N-S equations. First, since the mass of each particle is constant in the simulation, the problem of mass conservation is eliminated. Besides, the complicated convective acceleration $\frac{\partial v}{\partial t} + v \cdot \nabla v$ can be replaced by a simple differentiation of $\frac{dv}{dt}$ in a particle system.

The three terms on the right side of (5.8) represent the contributions of external body force, pressure, and viscosity. In conventional fluid dynamics, the last two terms are very difficult to calculate because it involves the computation of many partial differential equations. In SPH method, they can be easily solved by using the derived smooth field A_s, which is given by,

$$f_i^{pressure} = -\sum_j m_j \frac{p_i + p_j}{2\rho_j} \nabla W(|x - x_i|) \quad , \tag{5.9}$$

$$f_i^{viscosity} = \mu \sum_j m_j \frac{v_j - v_i}{2\rho_j} \nabla^2 W(|x - x_i|) \quad , \tag{5.10}$$

where $f_i^{pressure}$ and $f_i^{viscosity}$ are the force contributions from pressure and viscosity respectively. SPH computes pressure and viscous force from weighted contributions of neighboring particles rather than by solving linear systems of equations, and hence, the cost of computation is reduced.

As a novel meshfree particle method, SPH not only uses the Lagrangian particles as the computational frame for interpolation or differencing, but also uses the particles to carry the material properties. Because the collective movement of those particles is similar to the movement of liquid or gas flow in physics-based simulation, they can be modeled by the governing equations of the classical Newtonian hydrodynamics.

The discretization of the governing equations is based on these discrete particles, and the proposed particle-based formulations have been used to calculate the local density, velocity and acceleration of the fluid. The fluid pressure is calculated from the density using isotropic pressure model shown in Section 5.3, and the particle acceleration is then calculated from the pressure gradient and the density. For viscous flows, the effects of physical viscosity on the particle accelerations can also be included. As a Lagrangian particle method, SPH conserves mass exactly. In SPH, there is no explicit interface tracking for multiphase flows because the motion of the fluid is represented by the motion of the particles, and fluid surfaces or fluid-fluid interfaces move with particles representing their phase defined by the initial conditions.

5.2 Modified finite particle method

The Finite Particle Method (FPM) was first proposed in 2005 in [173]. FPM is also a meshfree method which uses the integral form of function representation based on a transformed Taylor series expansion, and the parameters in the shape function are approximated by a set of basis functions.

In FPM, particles with lumped volumes are used to represent the state of a system. The particles can either be fixed in an Eulerian frame as interpolation points or can move in a Lagrangian frame as material particles. A set of basis functions are employed in FPM to approximate a field variable and its derivatives through solving a pointwise matrix equation. Therefore, FPM is a meshfree, Lagrangian and particle method with desired accuracy depending on the order of the derivatives retained in the Taylor series expansion process.

For a particle system in a problem domain Ω, FPM lumps the fluidic property of a subsection on the corresponding particles (this is different from a grid-based method, where all the calculations are performed on top of the meshed grid). If we apply Taylor's expansion on a particular point $x_{(x1,x2,x3)}$ in three-dimensional space, a kernel function $f(x)$ at x could be expressed as,

$$\begin{aligned} f(x) = f(x_i) + \left\{ \frac{\partial f(x_i)}{\partial x_1} \quad \frac{\partial f(x_i)}{\partial x_2} \quad \frac{\partial f(x_i)}{\partial x_3} \right\} (x - x_i) \\ + \frac{1}{2}(x - x_i)^T \left\{ \begin{matrix} \frac{\partial^2 f(x_i)}{\partial x_1^2} & \frac{\partial^2 f(x_i)}{\partial x_1 \partial x_2} & \frac{\partial^2 f(x_i)}{\partial x_1 \partial x_3} \\ \frac{\partial^2 f(x_i)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x_i)}{\partial x_2^2} & \frac{\partial^2 f(x_i)}{\partial x_2 \partial x_3} \\ \frac{\partial^2 f(x_i)}{\partial x_3 \partial x_1} & \frac{\partial^2 f(x_i)}{\partial x_3 \partial x_2} & \frac{\partial^2 f(x_i)}{\partial x_3^2} \end{matrix} \right\} (x - x_i) + R_n(x - x_i), \end{aligned} \tag{5.11}$$

where $R_n(x - x_i)$ represents the remaining terms of Taylor's expansion. Equation (5.11) can be rewritten as,

$$f(x) = f(x_i) + \nabla f(x_i)^T (x - x_i) + \frac{1}{2}(x - x_i)^T \nabla^2 f(x_i)(x - x_i) + R_n(x - x_i). \tag{5.12}$$

Instead of using the conventional way of multiplying an additional weight function, the kernel function $W(|x - x_i|, h)$ is directly multiplied on both sides of Equation (5.12), and integrate it over the problem space Ω, and hence, the following equation could be obtained,

$$\begin{aligned} \int_\Omega f(x)W(|x - x_i|, h)dx = f(x_i) \int_\Omega W(|x - x_i|, h)dx + \nabla f(x_i) \int_\Omega (x - x_i)W(|x - x_i|, h)dx \\ + \frac{\nabla^2 f(x_i)}{2} \int_\Omega (x - x_i)^T W(|x - x_i|, h)(x - x_i)dx. \end{aligned} \tag{5.13}$$

The conventional FPM uses second-order kernel functions to approximate the function value, first derivatives and second derivatives. In order to achieve a more accurate blood flow simulation,the quartic kernel function is used to approximate the respective parameters, where,

$$W(|x-x_i|,h) = \frac{15}{7\pi h^3}\begin{cases} \frac{2}{3}-\frac{8}{9}(\frac{|x-x_i|}{h})^2+\frac{19}{24}(\frac{|x-x_i|}{h})^3-\frac{5}{32}(\frac{|x-x_i|}{h})^4 & \text{if } 0 \leq |x-x_i|/h \leq 2 \\ 0 & \text{if } |x-x_i|/h > 2. \end{cases} \tag{5.14}$$

Since the points distributed in the problem space are particles, each occupying individual an lumped volume ΔV_i, Equation (5.13) can be numerically approximated by summation over the N particles surrounding point x_i as follows,

$$\begin{aligned} &\sum_{j=1}^{N} f(x_j)W\Delta V_j = f(x_i)\sum_{j=1}^{N} W\Delta V_j + \nabla f(x_i)\sum_{j=1}^{N}(x-x_i)W\Delta V_j \\ &+\frac{1}{2}\nabla^2 f(x_i)\sum_{j=n}^{N}(x-x_i)^T W(x-x_i)\Delta V_j, \end{aligned} \tag{5.15}$$

where,

$$\Delta V_j = \frac{m_j}{\rho_j}. \tag{5.16}$$

For a point $x = \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix}$ in three dimensional space, the required number of function value, first derivatives and second derivatives are 1, 3 and 6 respectively since the matrices of first and second derivatives are symmetric. Therefore, it requires 10 basis equations similar to that of Equation (5.15) to approximate these terms. A set of 10 basis equations can be formed using $W, W_{x_1}, W_{x_2}, W_{x_3}, W_{x_1x_1}, W_{x_1x_2}, W_{x_1x_3}, W_{x_2x_2}, W_{x_2x_3}$ and $W_{x_3x_3}$ of the quartic kernel function.

Equation (5.15) is not computationally efficient for physics-based simulations since the matrix needs to be timestepped at 24fps. One approximation method to reduce the computational cost is to consider only the first order derivative of (5.11) in the sense of Taylor expansion. Hence, the computation of the terms in N-S equations which require only first order accuracy could be optimized. The discretization of $[f(x_i) \quad \nabla f(x_i)]^T$ could be written as,

$$\begin{Bmatrix} f(x_i) \\ \nabla f(x_i) \end{Bmatrix} = \begin{Bmatrix} \sum\limits_{j=1}^{N} \frac{m_j}{\rho_j} W & \sum\limits_{j=1}^{N} \frac{m_j}{\rho_j}(x_j - x_i) W \\ \sum\limits_{j=1}^{N} \frac{m_j}{\rho_j} \nabla W & \sum\limits_{j=1}^{N} \frac{m_j}{\rho_j}(x_j - x_i) \nabla W \end{Bmatrix}^{-1} \times \begin{Bmatrix} \sum\limits_{j=1}^{N} \frac{m_j}{\rho_j} f(x_i) W \\ \sum\limits_{j=1}^{N} \frac{m_j}{\rho_j} f(x_i) \nabla W \end{Bmatrix}, \tag{5.17}$$

where $W, W_{x_1}, W_{x_2}, W_{x_3}$ are used to form a group of four basis equations to approximate $f(x)$ and $\nabla f(x)$ in Equation (5.17) similar to that of Equation (5.15).

5.3 Equation of motion and discretization

As it has been discussed in section 5.1, the continuum Navier-Stokes in Equation (5.7) and (5.8) can be simplified using meshfree method, where the conservation of mass and conservation of momentum can be expressed as,

$$\frac{D\rho}{Dt} = -\rho \frac{\partial v_b}{\partial x_b} \quad , \tag{5.18}$$

$$\rho \frac{D v_a}{Dt} = \frac{\partial \sigma_{ab}}{\partial x_b} + F_e \quad , \tag{5.19}$$

and the scalar density ρ, the velocity component v_a, external force F_e and the total stress tensor σ_{ab} are dependent variables, while the spatial coordinates x_b and time t are independent variables. a and b represent the coordinate directions of the vector in the 3D space.

The stress tensor representation of N-S equations is employed instead of viscosity and damping ratio in this simulation. The summations in (5.18) and (5.19) are taken over repeated indices, while the total time derivatives are taken in the moving Lagrangian frame.

The total stress tensor σ_{ab} consists of two parts, isotropic pressure p and viscous stress τ, and it can be expressed as,

$$\sigma_{ab} = -p\delta_{ab} + \tau_{ab}, \tag{5.20}$$

where δ is the Kronecker delta, and $\delta_{ab} = 1$ if $a = b$, and $\delta_{ab} = 0$ if $a \neq b$ in fluid mechanics. The viscous stress of Newtonian fluids is generally proportional to the strain rate ε which is determined by the dynamic viscosity μ,

$$\tau_{ab} = \mu\varepsilon_{ab}, \tag{5.21}$$

where,

$$\varepsilon_{ab} = \frac{\partial v_b}{\partial x_a} + \frac{\partial v_a}{\partial x_b} - \frac{2}{3}(\nabla \cdot v)\delta_{ab}. \tag{5.22}$$

There are various methods which could be used to model the isotropic pressure in a computationally efficient manner. In this work, the choice is based on [174],

$$p = p_0((\frac{\rho}{\rho_0})^\gamma - 1), \tag{5.23}$$

where p_0 is the initial pressure, ρ_0 is the reference density, and γ is taken to be 7. Since the blood flow is modeled as a two-phase fluid, the quality averaged density and dynamic viscosity is then derived based on the void fraction,

$$\rho = \rho_p(1-\varsigma) + \rho_g\varsigma, \tag{5.24}$$

$$\mu = \mu_p(1-\varsigma) + \mu_g\varsigma, \tag{5.25}$$

where ς is the void fraction, ρ_p is the plasma density (1060 kg/m^3), ρ_a is the gas-phase density (1.205 kg/m^3), μ_p is dynamic viscosity of plasma at $37°C$ (3.56×10^{-3} $Pa \cdot s$), and μ_g is the dynamic viscosity of gas-phase at $37°C$ (18.6×10^{-6} $Pa \cdot s$).

Hence, for a particle i in the system, the equation of motion can be obtained as

$$(\frac{D\rho}{Dt})_i = -\rho_i \frac{\partial v_b}{\partial x_b} \quad , \tag{5.26}$$

$$(\frac{Dv_a}{Dt})_i = -\frac{1}{\rho_i}(\frac{\partial p}{\partial x_a}) + \frac{\mu}{\rho_i}(\frac{\partial \varepsilon_{ab}}{\partial x_b}) + F_e \quad , \tag{5.27}$$

The discretization of structure position was implemented using the scheme described in $Algorithm 1$. In each time step, the position and velocity vector of a particle was firstly updated to time level $t + \delta t$. The boundary condition would be updated based on the evaluation of particles and scene geometry in the problem space at time t. Next, a particle searcher was used to construct the domain of neighboring particles. The particle velocity and fluid field would then be updated using the proposed meshfree method. For example, the change of fluid density can be derived by computing the first-order derivative of velocity using the FPM described in Equation (5.17) (Simply change f to v). This process would be repeated recursively until the end of the simulation. The time step is chosen to be 0.01s. In the meshfree method, the time step can be chosen to be much larger than the requirement of Courant–Friedrichs–Lewy (CFL) condition. It has been shown in [174] that 0.01s is sufficient to converge the simulation result with the analytical series solution.

Algorithm 1 : Discretization of the hemodynamic simulation in one time step

```
1. Update particle positions: $x_i(t+\delta t) = x_i(t) + \delta t \cdot v_i(t+\delta t/2)$
2. Set boundary conditions
3. Search for neighboring particles
4. Update particle velocities: $v_i(t+\delta t/2) = v_i(t-\delta t/2) + \delta t D v_i(t)$
5. Solve for fluid density: $\rho_i(t+\delta t/2) = \rho_i(t-\delta t/2) + \delta t D\rho_i(t)$   $\frac{D\rho}{Dt} = -\rho\frac{\partial v^\beta}{\partial x^\beta}$
6. Evolve by time step $\delta t$
7. Repeat
```

5.4 Boundary condition

For hemodynamic simulations based on meshed geometry such as FEM and FLIP, boundary conditions can be well defined using the continuum equations, and the corresponding parameters can be obtained based on assumed velocity profile. Many approximation strategies have been developed to model the fluidic trajectory at wall boundary such as Dirichlet boundary conditions and Neumann boundary conditions [175, 176]. Nevertheless, FPM and SPH are meshfree methods where the boundary condition only depend on the force interactions between particles defined by particle-particle distance. Modeling of such boundary conditions is not straightforward since there is no pre-defined velocity profile in the near-wall area. Besides, the fluidic parameters of each particle needs to be directly computed based on the contributions of neighboring particles in meshefree methods, and the conventional way of assigning viscosity and density values on each meshed-grid no longer applies.

In fact, accurate modeling of the boundary conditions used to be regarded as a major defect of meshfree methods. A proper modeling of boundary condition for blood circulation should enable the arterial wall to render sufficient repulsive force to prevent penetration of fluid particles, and the shear force for the drag reduction of fluid. In the past decade, there are many works which attempted to model the boundary condition in a meshfree manner. A commonly used approach is by seeding ghost or virtual particles on the surface of the boundary with frozen positions. When a mobile fluid particle approaches within a certain range of these ghost particles, a repulsive force would be generated and applied on the corresponding particle so that the particle can be confined within the volume as shown in Figure 5.2.

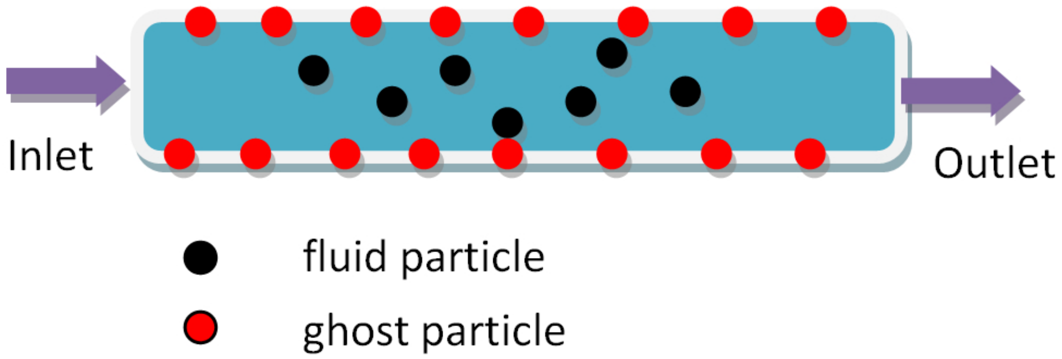


Fig. 5.2 Modeling of boundary condition

In general, it is feasible to use virtual particles to implement the solid boundary conditions. Common choices of virtual particles could be classified into two categories: virtual particles that are located right on the solid boundary and virtual particles that are outside the boundary. In this work, the virtual particles are seeded outside the boundary in order to maintain a specific range of clearance. The repulsive force of these models is in Lennard-Jones 6-12 forms which have been applied in many simulations of the dynamics of molecules.

A specific requirement of hemodynamic simulation is that a portion of the fluid particles needs to be absorbed by the blood vessel at some velocity-dependent rate, so that pharmacological treatments could become effective on the patient. This is the main reason to seed the virtual particles outside the boundary. The absorption is modeled in a way that when a fluid particle enters the defined clearance range of boundary with a certain velocity, this particle and the associated volume would be absorbed by the arterial. The particle volume could be programmed as any desired shape. This feature is important in medical application since an appropriate absorption rate for each patient is highly personalized, and accurate control of the absorption rate is importatnt for the technical success of the treatment.

Considering the domain of blood flow in patient-specific vascular system, ghost particles are seeded on the surface of the blood vessel with frozen positions [150]. Suppose particle *i* is a boundary particle. All the other particles within its support, $N(i)$, can be divided into three subsets:

(1) $I(i)$: all the interior points that are the neighbors of i;

(2) $B(i)$: all the boundary points that are the neighbors of i;

(3) $G(i)$: all the exterior points that are the neighbors of i, in other words, all the ghost particles.

Therefore, $N(I) = I(i) \bigcup B(i) \bigcup G(i)$ form the basic structure of all boundary particles. When a mobile fluid particle approaches a certain range of these ghost particles, a repulsive force will be generated to interact with the particle. Hence, a physics-based construction of boundary condition in a meshfree manner is defined.

5.5 Results and discussion

5.5.1 3D simulation of blood flow

The proposed SPH method was implemented in a simulation of injecting blood flow in a reconstructed patient-specific artery model. The computing platform was a Intel(R) Core(TM)i5-3470 CPU @ 3.2GHz, 8G RAM with NVIDIA GeForce 605. Figure 5.3 shows the 3D simulation of blood flow in a physics-based manner. The arterial wall was rendered in transparent color so that clinicians can visualize the fluid motion inside the blood vessel. This color effect can be turned off based on the user preference. The parameters for the simulation of blood flow is shown in Table 5.1.

The blood was injected from the top part of blood vessel through a small incision, and an injection needle was also modeled in the simulation. Injection of continuous blood flow (or drug fluid) into the human body is an important endoscopic surgical procedure. As it can be seen in the result, a void was observed at the beginning of the simulation. This is due to the gas-phase fluid in human blood modeled by the gas particles. The void gradually dissipated

Table 5.1 Flow parameters of 3D blood flow simulation

Parameter	Value
Plasma density	1060 kg/m^3
Gas density	1.205 kg/m^3
Plasma viscosity	3.56 $\times 10^{-3} Pa \cdot s$
Gas viscosity	1.86 $\times 10^{-5} Pa \cdot s$
Void fraction	20%
Particle generation rate	300/s
Gauge pressure	5332 Pa (40mmHg)

as the blood flows through the blood vessel from Figure 5.3(a) to (h). This phenomenon compares well with the experimental results shown in the literatures [177, 178]. The blood flow was confined in the problem space by the boundary formed by ghost particles. In order to demonstrate the simulation of free surface fluid, the blood which flew out of the blood vessel was accumulated in an ‘invisible’ box at the bottom. Figure 5.3(h) shows the accumulation of free surface blood fluid.

In this simulation, important fluidic phenomena include viscous flow, void forming/dissipation and free surface fluid have been achieved. Fluid parameters such as viscosity, density, and color could be customized by users. Computer graphics simulation of blood flow provides an intuitive solution to the clinicians to visualize the dynamics of blood flow in patient’s vascular system. Such simulation is meaningful if it could be performed in patient-specific models. Conventional way of using generic drift flux model can be very accurate in predicting the dynamics of the fluid, however, it does not complies with the geometry of patient’s blood vessel. By changing the boundary conditions and updating the reconstructed model geometry based on the medical requirements, this simulation can also be repeated on other patients so as to assist the clinician in diagnostics and surgical planning.

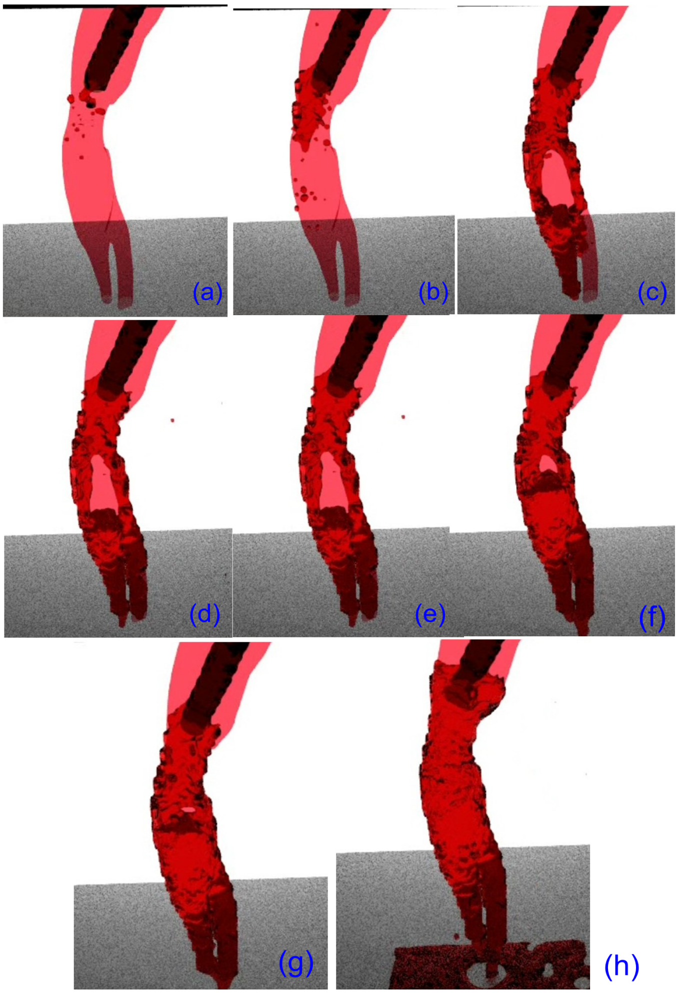


Fig. 5.3 3D simulation of blood flow in patient-specific blood vessel in 8 discrete time steps from (a) to (h). The blood flows out of the aorta is accumulated in an invisible box at the bottom, which is shown in (h)

5.5.2 Simulation of internal fluidic structure

Interventional radiology is carried out through a small anatomical opening, where all surgical procedures need to be monitored by a probe inserted inside the blood vessel. Due to the low-light condition and the limited resolution of the probe camera, the vision of anatomical environment is quite restricted to the clinician during the treatment. Manipulation of the probe location and adjustment of focal length are generally difficult especially for junior clinicians. Therefore, simulation of the internal fluidic structure of patients in a virtual endoscopic vision is important.

Figure 5.4 shows the simulation results of the internal fluidic structure of a human vascular system. A virtual camera is modeled inside the blood vessel. In practice, the focal length of the camera can be adjusted by the clinician. In this simulation, both focused and unfocused visions are readily available for comparison. The cytoskeleton of the blood cell is modeled as a compressed sphere in vertical directions. As it is shown in the result, when the camera was not focused (first row of Figure 5.4), there were a significant amount of noises which could affect the clinician's vision during the treatment. These noises resembled the optical noises that happened in the real surgery. When the probe was focused properly by adjusting the focal length (shown in the second row of Figure 5.4), most of the noises were eliminated and the vision through probe became clean and sharp. According to the feedback from the collaborating clinician in the National University Hospital (Singapore), the adjusting of probe focal length normally takes more than half an hour to produce clear images during vascular surgeries. This simulation could be used as a training scheme for junior surgeons, offering them a realistic virtual reality environment to acquire the respective surgical skills, and hence, the potential surgical risk could be reduced.

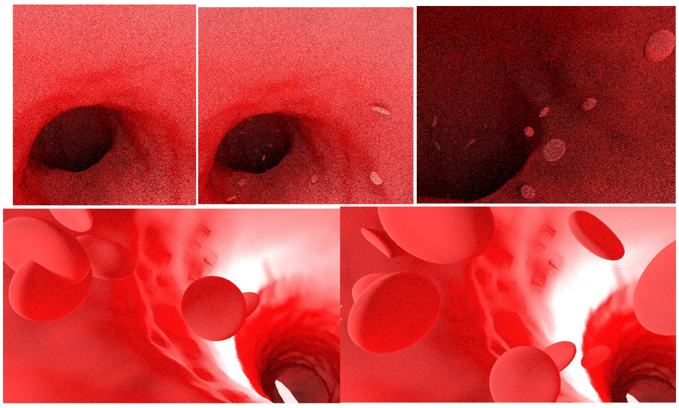


Fig. 5.4 Simulation of internal fluidic structure of blood flow

5.5.3 Simulation of hand and heart circulation

Two sets of patient medical images were used to evaluate the performance of the proposed FPM method. These data were acquired from the National University Hospital, Singapore. The computing platform was a Intel(R) Core(TM)i5-3470 CPU @$3.2GHz$, 8G RAM with NVIDIA GeForce 605. The information of medical images is shown in Table 5.2.

Table 5.2 Dataset specifications of heart and hand circulation

Dataset	Hand	Heart
Bits allocated	8	8
Raw × Column	512 × 512	192 × 156
Slices	136	27
Pixel size (mm)	0.39 × 0.39	1.67 × 1.67
Inter-slice spacing	4.0	8.0
Size (MB)	171.25	15.42

Figure 5.5 shows the blood flow circulation of hand with respect to the velocity profile. The flow parameters for the simulation of hand circulation are shown in Table 5.3. The arrows indicated the flow trajectory predicted by the dynamic medical images. This is important in hemodynamics simulation because it avoids the time-consuming computation of diverge decision at each junction of vascular system, and more closely resembled the patient's flow direction. It can be seen that the largest amount of blood flow was delivered to the thumb, while the last finger has the least amount of delivered blood flow. This result converges with the fact and explains why the thumb is the strongest among all fingers. Two blockages in the blood circulation were simulated in this part. It can be seen in Figure 5.6 that the blockages induced narrowing in the artery, and therefore, a significant increase in velocity was observed around the blockages.

In order to perform quantitative analysis of the change of flow velocity due to the blockage growth in blood vessel, the cross-sectional velocity profiles of blood flow 6 mm before and after the blockage A was investigated. As it can be seen in Figure 5.7, the maximum flow

Table 5.3 Flow parameters for the simulation of hand circulation

Parameter	Value
Plasma density	1060 kg/m^3
Gas density	1.205 kg/m^3
Plasma viscosity	3.56 $\times 10^{-3} Pa \cdot s$
Gas viscosity	1.86 $\times 10^{-5} Pa \cdot s$
Void fraction	10%
Particle generation rate	100/s
Gauge pressure	667 Pa (5mmHg)

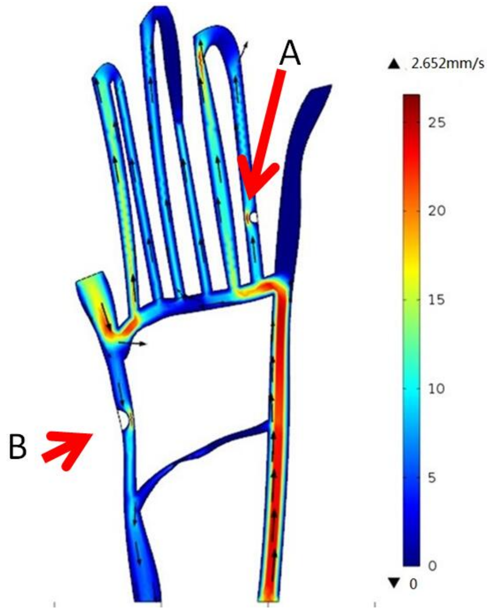


Fig. 5.5 Velocity profile (mm/s) of blood flow simulation in hand circulation. Two blockages (A and B) were simulated in the work

velocity at the blockage is almost three times larger than blood flow before and after the blockage. The increase in flow velocity implies the increase in flow pressure, and the accumulative effect of pressure increase in one particular position of artery could possibly cause pain and overstrain in the fingers. A vascular blockage in hand circulation is considered as a type of Peripheral Artery Disease (PAD). Continuous growth of the blockage can lead to a decrease in blood flow. Complete blockage is the pathogen of gangrene and the symptoms can include redness, skin warm to the touch, and swelling [179, 180].

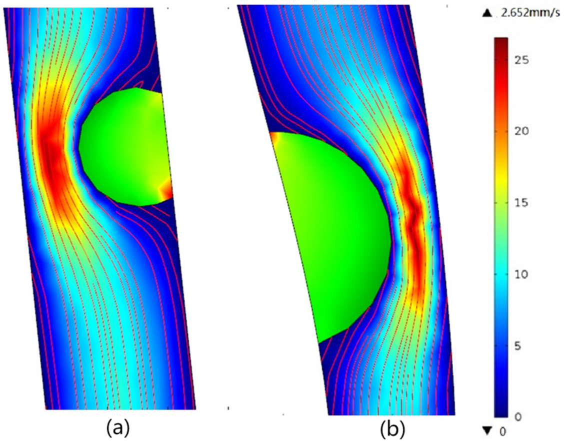


Fig. 5.6 Velocity profile around the two blockages in the hand circulation (a) the blockage located at the bottom left (b) the blockage located at the upper right

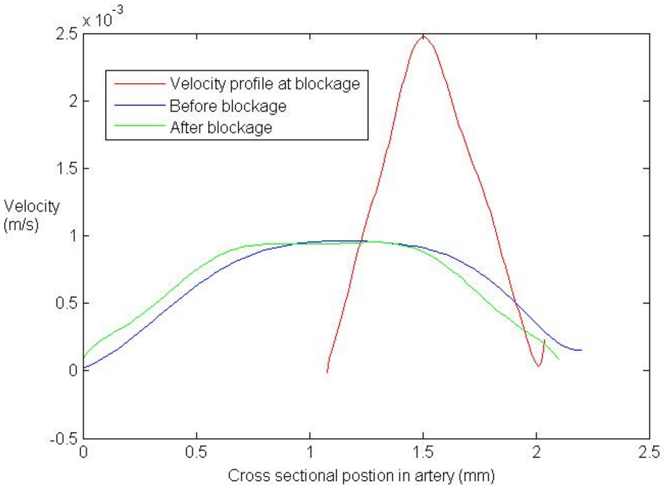


Fig. 5.7 Blood flow velocity in the simulation of hand circulation 6mm before/after and at the blockage A

Figure 5.8 shows the blood flow circulation of heart with respect to velocity profile. Table 5.4 shows the flow parameters of the simulation. In human vascular system, blood was continuously pumped out from the Left Ventricle (LV) and pumped into Right Atrium (RA) in every pulse. This simulation isolated the blood flow in one pulse without considering the influence from the next pulse. Narrowing of artery due to the growth of blockage was simulated in this work. Numerical analysis of the velocity profile around the blockage was performed to assist the diagnostic evaluation and surgical planning of artery disease. As it can be seen in Figure 5.9, the maximum flow velocity in the narrowing part is about 2.5 to 3.1 times larger than the velocity 30mm before and after the narrowing part. The narrowing of artery in heart circulation could reduce the transfer of oxygen-rich blood to the heart of the patient. Without enough blood supply, the patient's heart becomes starved of oxygen and the vital nutrients. Early development of artery narrowing may not induce clear symptoms, however, severe development of the disease could cause angina and chest pain. Heart attack may also occur if the energy demands of the heart become much larger than the blood supply.The simulation proposed in this work could be helpful for clinicians to analyze the pathological condition of patients in an efficient and risk-free manner.

Table 5.4 Flow parameters for the simulation of heart circulation

Parameter	Value
Plasma density	1060 kg/m^3
Gas density	1.205 kg/m^3
Plasma viscosity	3.56 $\times 10^{-3} Pa \cdot s$
Gas viscosity	1.86 $\times 10^{-5} Pa \cdot s$
Void fraction	16%
Particle generation rate	120/s
Gauge pressure	6666 Pa (50mmHg)

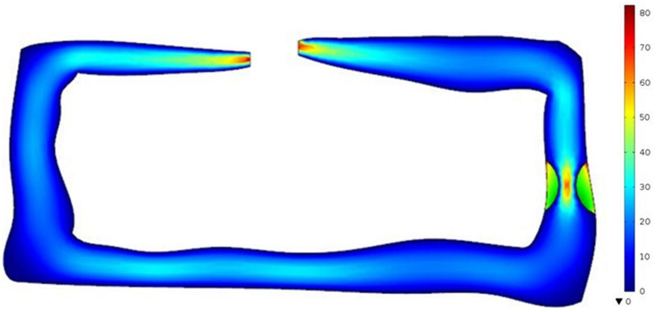


Fig. 5.8 Velocity profile (mm/s) of blood flow simulation in hear circulation.

In current study of pathology, the hemodynamic changes occuring around the blockage in blood vessel are poorly understood. According to the medical report in [181, 182], most patients (90%) will recover without clinical evidence of hemodynamic insufficiency.

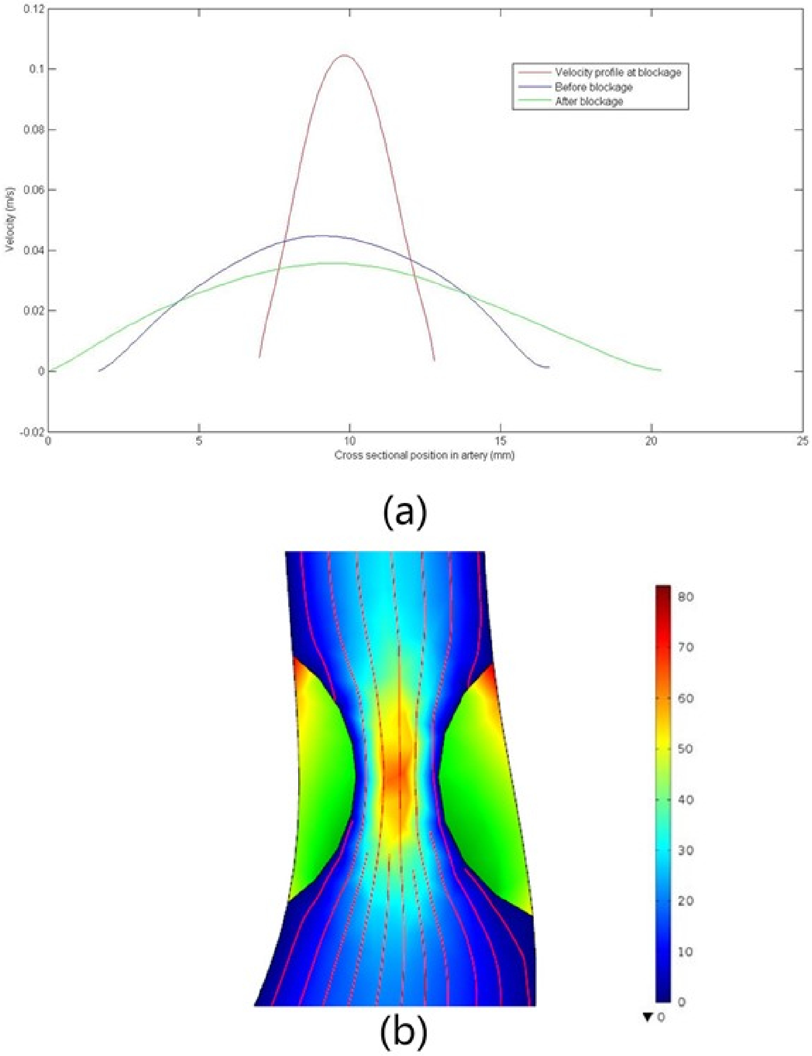


Fig. 5.9 Velocity profile of blood flow simulation in heart circulation 30mm before/after and at the narrowing of artery (a) Numerical results (b) Streamline representation

The proposed meshfree method for hemodynamic simulation in heart and hand circulation presents a new attempt to evaluate the influence of blockage in artery. On the other hand, the results closely resembled many physiological behavior of natural human blood circulation, such as the distribution of blood supply in hand and the pumping of blood flow from LV.

Modeling with meshfree method only requires a set of unstructured points that cover the domain of interest. Since the method has simple topological data structures, it allows easy adaptive refinement, easy parallelization, and flexible interpolation in the problem space. Many problems that currently cannot be solved by finite element or finite difference methods are tractable by meshfree methods. Although much have been achieved in meshfree formulations, there are still many tasks and challenges remaining. Some main issues include the scalable implementation of essential boundary conditions, accurate nodal integration strategies, and stabilization schemes for discretized structure [183]. With further development of the algorithm and calibration based on experimental data, this work has the potential to be used as an efficient tool to assist surgical planing and evaluation.

Lack of quantitative experimental validation is perhaps the major deficiency of this work. A surgical simulator would not be convincing to users without test and calibration through on-site experiment. Due to safety considerations, a large amount of data needs to be acquired from clinical tests to converge with real-life surgical procedures. A series of experiments should be conducted before introducing the simulator into actual applications. However, such experiments could be costly and risky if they are applied on human or animal subjects.

5.6 Summary

This chapter introduced the application of the proposed SPH and FPM method in hemodynamic simulations. The 3D simulation yields good simulation realism based on visual inspection in an application of injecting human blood into blood vessel through a small anatomical opening, which resembles the surgical procedure in interventional radiology. Important phenomena such as viscous flow, back flow, void forming/dissipation and free surface fluid have been achieved in the simulation. The simulation of hand and heart circulation achieved satisfactory robustness in evaluating the outcome of blockage growth in blood vessel with respect to the velocity of blood flow. The use of dynamic medical images helps to predict the flow trajectory in the vascular system. The meshfree method conserves the mass and momentum in N-S equations, and thus, it is very computationally efficient in simulating complex fluidic behavior such as blood circulation. It is observed that a dramatic increase in blood flow velocity will be induced when a blockage occurred in the artery. Narrowing of the blood vessel due to blockages is the etiology of many severe vascular diseases such as abdominal aortic aneurysm and embolism.

Chapter 6

Simulation of Vascular Drug Delivery

Developing patient-specific model for 2-Dimensional (2D) fluidic simulation of chemotherapy drug delivery is important in medical application. Investigation of flow trajectory and evaluation of flow regime in the sense of controlled drug injection is crucial for the technical success of the treatment. This chapter proposes a two-phase fluidic method to simulate drug delivery in 2D patient-specific model. The patient-specific hepatic vascular model with unit-width skeleton is obtained from the results shown in Chapter 4.1. Flow regime of the drug in vascular system is identified through a designed experiment. A two-phase fluidic model based on bubbly flow regime is proposed for the drug delivery simulation.

6.1 Flow regime identification and characterization

It has been reported since 1983 that chemoembolization improves survival of patients with unresectable hepatocellular carcinoma (HCC) [184]. This therapeutic method comprises two procedures: local delivery of chemotherapy, and a minimally invasive procedure called embolization. In the first procedure, the anti-cancer drug is designed to be delivered to the cancerous tumor at an appropriate flow rate, which is monitored by MRI fluoroscopy.

Two-phase flows can be classified into four basic flow regimes [185]. In general, a gradual increment in the void fraction consecutively produces the transition between different flow patterns including bubbly, cap-bubbly, slug and churn-turbulent. The void fraction only provides a guide to determine the flow regime, and the precise definition is strongly subjective relying on linguistic description and graphical illustration.

Characterizations of bubble size and bubble rate are important for both medical application and theoretical investigation. Most of the existing works use arbitrarily assumed bubble rate and size with no regard to the actual void fraction in human blood [186, 187]. In accordance to medical report [188, 189], four levels including 5, 10, 15 and 20 ml/min of gas absorption rate were used to quantitatively investigate the bubble rate in vascular fluid in this work. The sizes of bubbles in aorta are normally less than 2.5mm, and generation of bubble in micron level is difficult. An experiment of using micron injection needle to control the bubble size and using high speed camera was designed to quantify the bubble rate. Four different sizes of bubbles were investigated with diameters varying from 1.29 to 2.05mm. The setup of the experiment is shown in Figure 6.1.

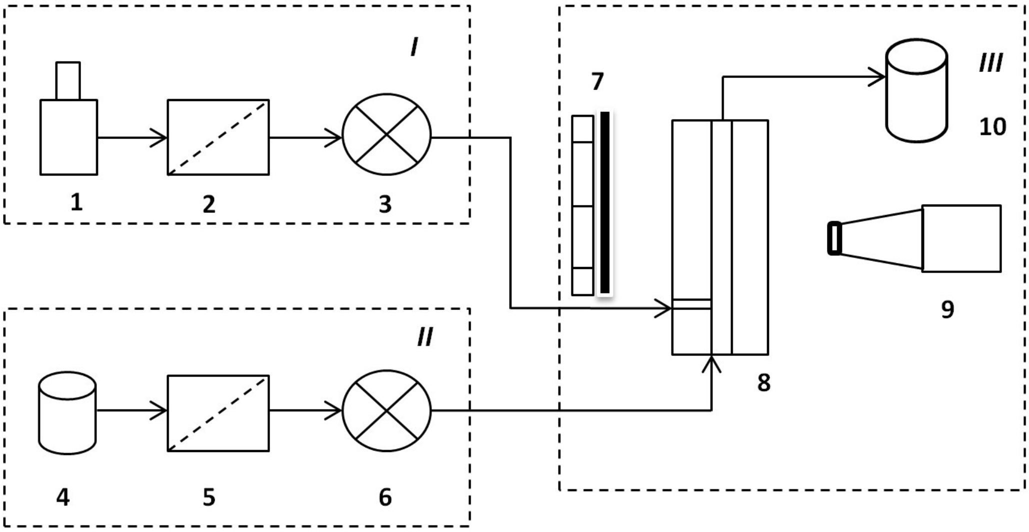


Fig. 6.1 Experimental setup. ***I*** Air supply system (1. air bottle, 2. check orifice, 3. syringe pump). ***II*** Liquid supply system (4. liquid reservoir, 5. check orifice, 6. syringe pump). ***III*** Test section (7. light source, 8. test channel, 9. high speed camera, 10. liquid reservoir)

Controlled injection of gas and liquid was carried out by using a medical syringe pump. A T-junction structure at the inlet of testing channel was used for bubble formation. When the injected two-phase flow became steady, the camera started to take images and record the dynamics of the fluid flow. It can be seen in Figure 6.2 that bubbles were generated at the tip of the injection needle when enough gas molecules were formed and the cluster of molecules was able to withstand diffusion from its surface.

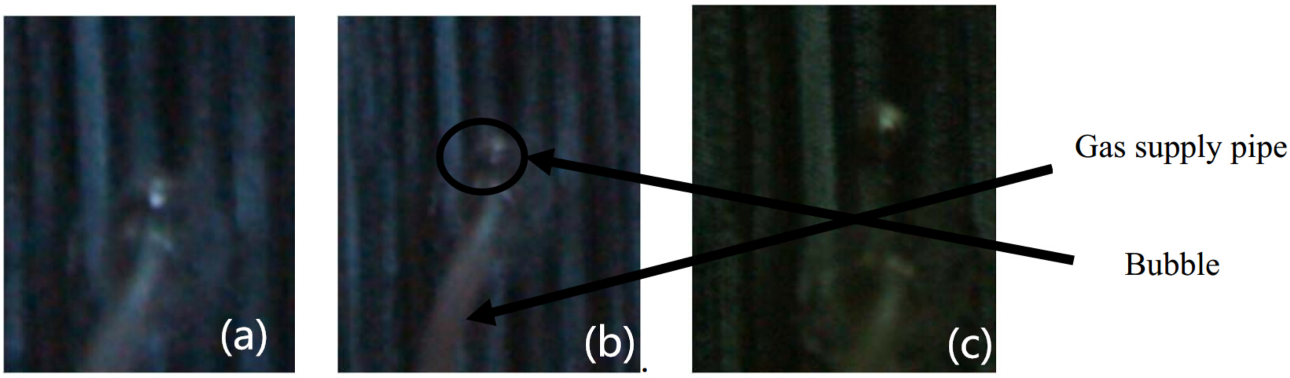


Fig. 6.2 Bubble generated at the tip of the injection needle. The injection needle is connected at the end of the air supply pipe and submerged 0.8mm into the testing channel

When the diameters of the injection needle were 0.8mm and 0.9mm, the bubbles were observed to detach and accelerate very quickly at the entry region, and very few bubbles coalesced. Once the steady state of the two-phase fluid has been achieved, it can be seen in Figure 6.3 that 99.2% of the bubbles are of the similar sizes with only a few big bubbles randomly positioned in the test channel. At this moment, the big bubbles occupied about $3-4\%$ of the total bubble volume. The kinetic energy of the small bubbles was much larger than big bubbles due to their high velocities, and the big bubbles after coalescence with small bubbles, were then accelerated. Thereafter, some big bubbles were still at a relatively low velocity and underwent obvious shape oscillations. They attracted the remaining uncoalesced bubbles nearby and the shape oscillated till they flowed out of the channel. Figure 6.4 demonstrates this phenomenon.

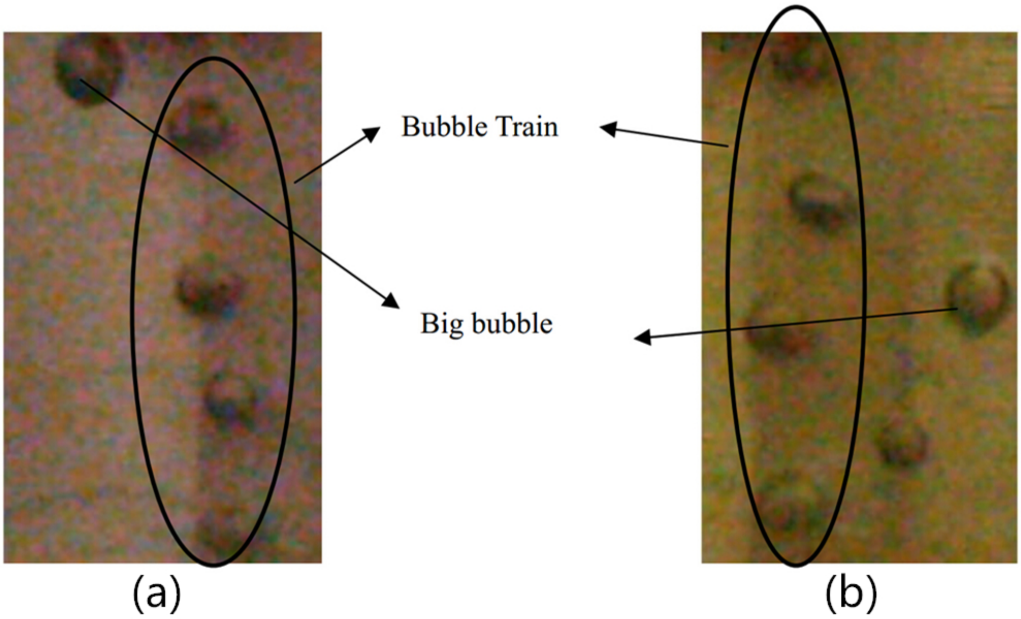


Fig. 6.3 Bubbles in steady state for needle size (a) 0.8mm and (b) 0.9mm.

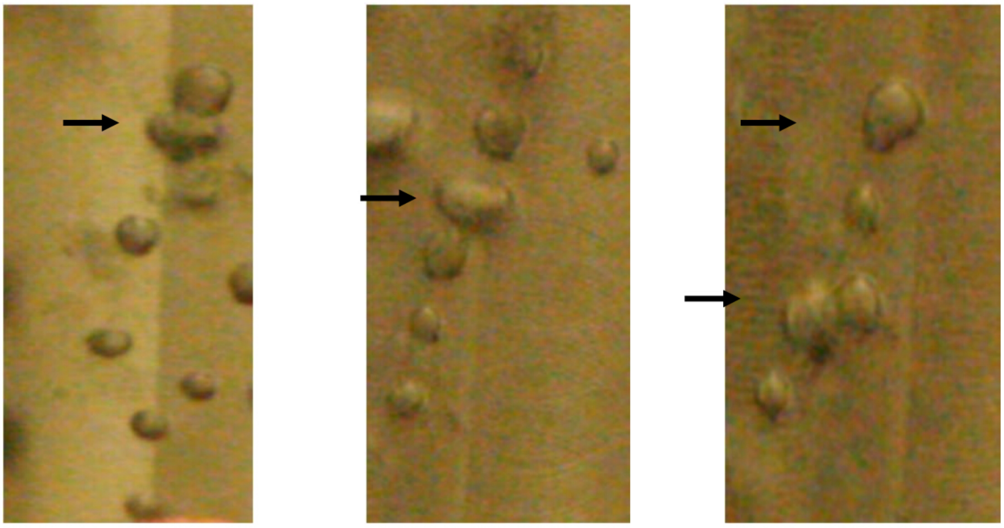


Fig. 6.4 Big bubble collide with the small bubbles. Coalescent occurs after the collision and the big bubbles is accelerated. The bubbles undergoing collision are marked by the solid arrows.

When the diameters of the injection needle were increased to 1.0mm and 1.2mm, the bubbles generated were much larger with slower velocities. Almost all (over 95% of the total amount of bubbles) bubbles would coalesce with other bubbles in the experiment. The flow trajectory and distribution of bubbles were quite random, and the shape oscillation phenomenon of bubbles occurred during the whole process of recording. Figure 6.5 shows the steady state fluid and bubbles.

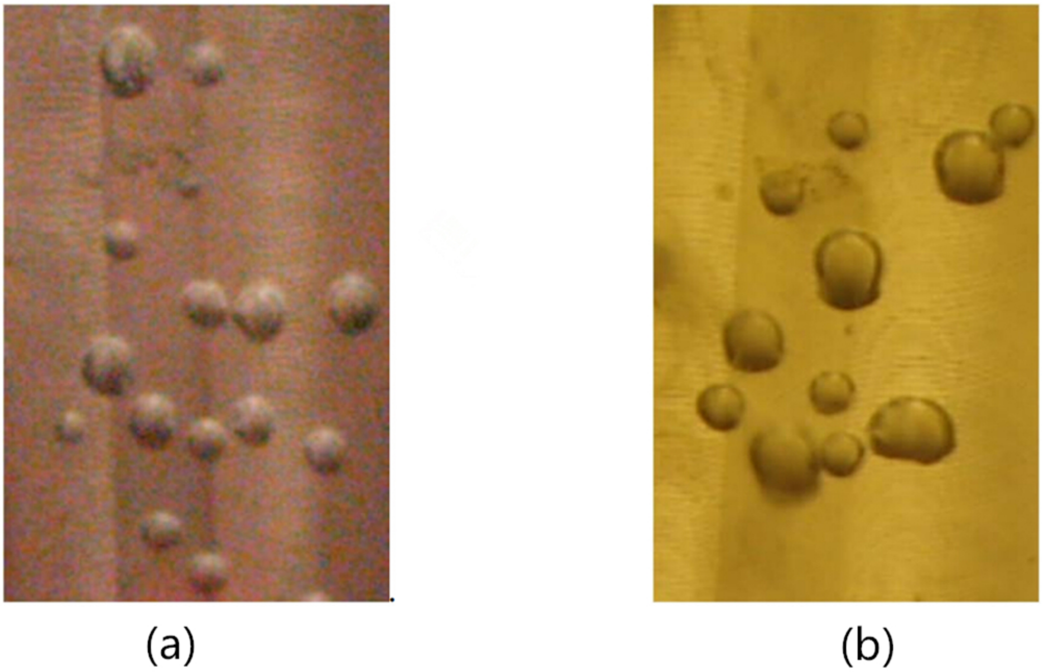


Fig. 6.5 Bubbles in steady state for needle size (a) 1 mm and (b) 1.2 mm.

Based on the observation in the above experiments, it can be seen that most bubbles are on the order of millimeter in size and there are generally no big bubbles which could choke the vessel. Hence, only bubbly flow pattern is considered in this work. The measured bubble rate and bubble size with respect to different gas absorption rates is shown in Figure 6.6. The result showed that the volume of gas is not conserved. This is due to the fact that a bubble can be formed only when a cluster of sufficiently large number of gas molecules is nucleated and able to withstand diffusion from its surface. The data is used as a reference to calibrate the fluidic simulation of drug delivery in this work.

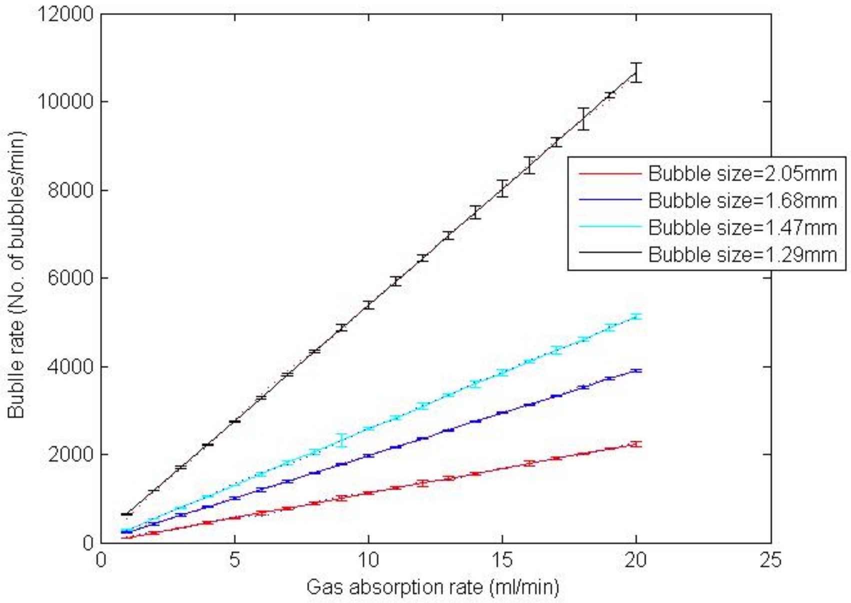


Fig. 6.6 Experimental result of bubble size and bubble rate with respect to the gas absorption rate. The *Solid lines* show the curve fitting result of the data, The *dashed lines* show the original measured result, and the *vertical bars* show the standard derivation.

6.2 System equation of vascular drug delivery

In this work, the vascular drug delivery is assumed to be steady, incompressible and Newtonian. As it has been discussed in Chapter 4, the geometry of the blood vessel is modeled as segments of conical pipe with continuously varying diameters. Different from the conventional way of simulating drug delivery as one-phase fluid, the fluid in this simulation is modeled as a two-phase porous medium system. In a two-phase flow, the total pressure drop is due to the variation of kinetic energy, and the kinematic pressure drop can be modeled using the Hagen-Poiseuille equation, which is given by,

$$\Delta P_k = \frac{8\mu L\dot{m}}{\pi R^4 \rho_h}, \tag{6.1}$$

where μ is the dynamic viscosity, L is the length of the tube, $\dot{m}$ is the mass flow rate, R is the radius of the tube, and ρ_h is the homogeneous density of the two-phase flow. The

Hagen-Poiseuille equation is a physical law that describes slow viscous incompressible flow through a constant circular cross-section. The homogeneous density ρ_h can be calculated using the void fraction of the fluid, which is given by Equation (6.2),

$$\rho_h = \rho_l(1-\vartheta)+\rho_g\vartheta, \tag{6.2}$$

where ϑ is the void fraction of the fluid. The quality averaged viscosity of the fluid can also be derived based on ϑ, which is given by

$$\mu = \vartheta\mu_g+(1-\vartheta)\mu_l, \tag{6.3}$$

The standard Hagen-Poiseuille model in Equation 6.1 is valid for a tube section with constant radius. However, it can be modified to apply to a conical tube with a linearly varying radius. Therefore, the blood vessel is considered as being made up of numerous thin sections. Each of the sections can be approximated as a constant radius tube section where the Hagen-Poiseuille equation is able to be used. For a general branch segment that is filled and represented as a conical pipe with radii R_1 and R_2 at its ends, the kinematic pressure drop can be derived as.

$$\Delta P_k = \int_0^L (\frac{8\mu\dot{m}}{\pi R^4\rho_h})dx, \tag{6.4}$$

$$R = \frac{R_2-R_1}{L}x+R_1, \tag{6.5}$$

$$\Delta P_k = \frac{8\mu L\dot{m}}{\pi\rho_h}\left[\frac{1}{3}\left(\frac{1}{R_1R_2^3}+\frac{1}{R_1^2R_2^2}+\frac{1}{R_1^3R_2}\right)\right]. \tag{6.6}$$

The two radii and the length of the segment are obtained in the vascular reconstruction step. Hence, by input the mass flow rate of drug injection, the partial length of each segment that is filled by fluid at each time step can be simulated.

6.3 Rendering method

Rendering of fluid in 2D space has been extensively studied by the fluid dynamic community in the past two decades. Early works of fluid rendering focus on using texture-based approaches such as the Volume Line Integral Convolution (LIC) method proposed by [190]. LIC method defines the value at each point in an output texture image as a weighted sum of the values that lie at evenly spaced intervals along the streamline. The streamline originates at the corresponding point in an input texture image and extents in both directions. The LIC method has been further developed by Cabral and his team who created streamline-like flow representations by filtering noise textures along the underlying vector field [191].

Another branch of methods for fluid rendering in computer graphics is the geometry-based methods. These methods render the flow motion directly using graphical icons. In [192], streamlines are depicted as simple, unshaded lines of equal thickness. [193] represents the streamline as 3-Dimensional (3D) geometrical objects, which could raise the number of polygon count in the scene. Dash tube has also been used as the rendering object in geometry-based methods. [194] depicted the streamline as dash tubes with an animated opacity texture. Nevertheless, most of these methods are constrained to be used only in the rendering of static fluids, which display the whole streamline or particle path as a one-time event.

The primary requirements for 2D rendering of drug delivery in medical applications are robustness and computational efficiency. Therefore, we based the rendering technique on the ‘virtual tubelets’ method proposed by Schirski and his team in [195]. Cylindrical billboard is used as the basis geometry in this method. The cylindrical billboards in the simulation are essentially a kind of quadrilaterals, which are aligned to face the viewer, and drawn or textured appropriately to create tubes in the final rendering. As it is shown in Figure 6.7, the head of the list of quadrilaterals is positioned at the current position of the corresponding particle with its tails passing through the particle's previous positions. It has been reported that the quadrilateral representation has advantages over the conventional ellipse representation in the aspect of rendering speed [196, 197].

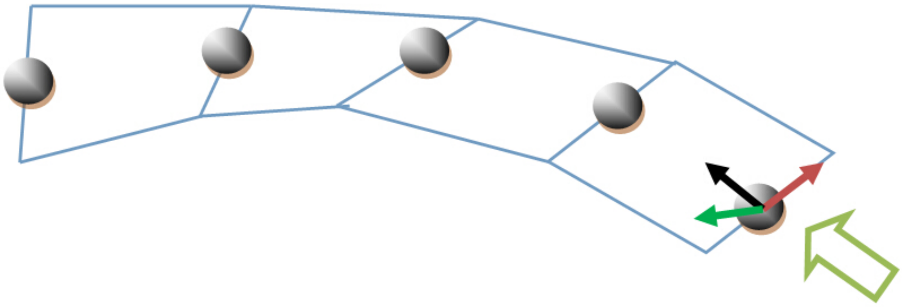
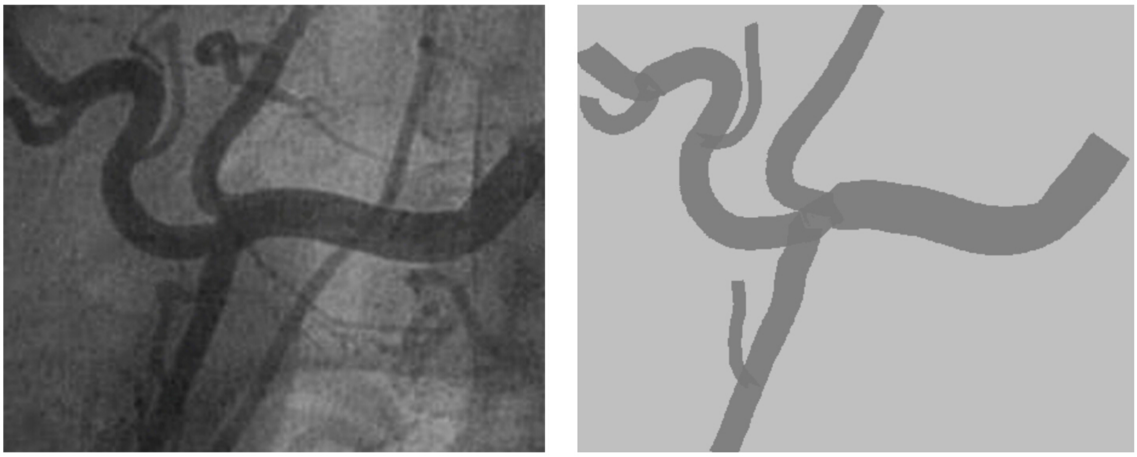


Fig. 6.7 Anatomy of a virtual tubelet. The black axis is the direction to the position in next time step. The green axis is the direction to the viewer.

In order to display the motion of drug fluid in hepatic vessels, the flow is rendered as overlaying and semitransparent quadrilaterals, which indicates the flow trails. The quadrilaterals could be drawn in decreasing grayness from the head to the last quadrilateral to depict the washing out effect during injection. For each tube segment, only one or two quadrilaterals are required to be drawn. Hence, the computational cost could be reduced compared to the conventional methods where numerous polygons are needed to display one segment.

6.4 Results and discussion

The simulation of drug delivery process in hepatic vessel is implemented using a NVDIA Quadro FX 4600 graphics card in an Intel Xeon CPU @3.00GHz. A comparison of reconstructed patient's vessel tree and the actual medical image is shown in Figure 6.9. The unit-width skeleton is obtained using the reconstruction and post-processing technique described in Chapter 4.1. The geometrical deformation of the blood vessel during the drug delivery is neglected in this simulation. It can be seen that the patient-specific hepatic vessel tree closely reassembled the geometry shown in the medical image. Therefore, the proposed fluidic simulation of drug delivery can be directly used to assist on-site diagnostics and surgical planning.



(a) (b)

Fig. 6.8 Comparison of (a) actual medical images and (b) reconstructed hepatic vessel tree.

Figure 6.9 shows the rendered 2D visualization of drug delivery in four discrete time steps using the proposed two-phase flow model. As it has been discussed in the previous section, the quadrilaterals which are aligned along the viewing direction are rendered to visualize the movement of drug fluid through the flow modeled using the proposed two-phase fluidic method. In medical applications, the physics-based flow model is unique and is indispensable for an accurate simulation of drug fluid.

Points A and B are marked as control points to calibrate the flow rate in the simulation. The result shows that the flow rate of drug fluid modeled by two-phase flow method has decreased by 8-9% compared with pure liquid fluid model. This decrease is generally lower

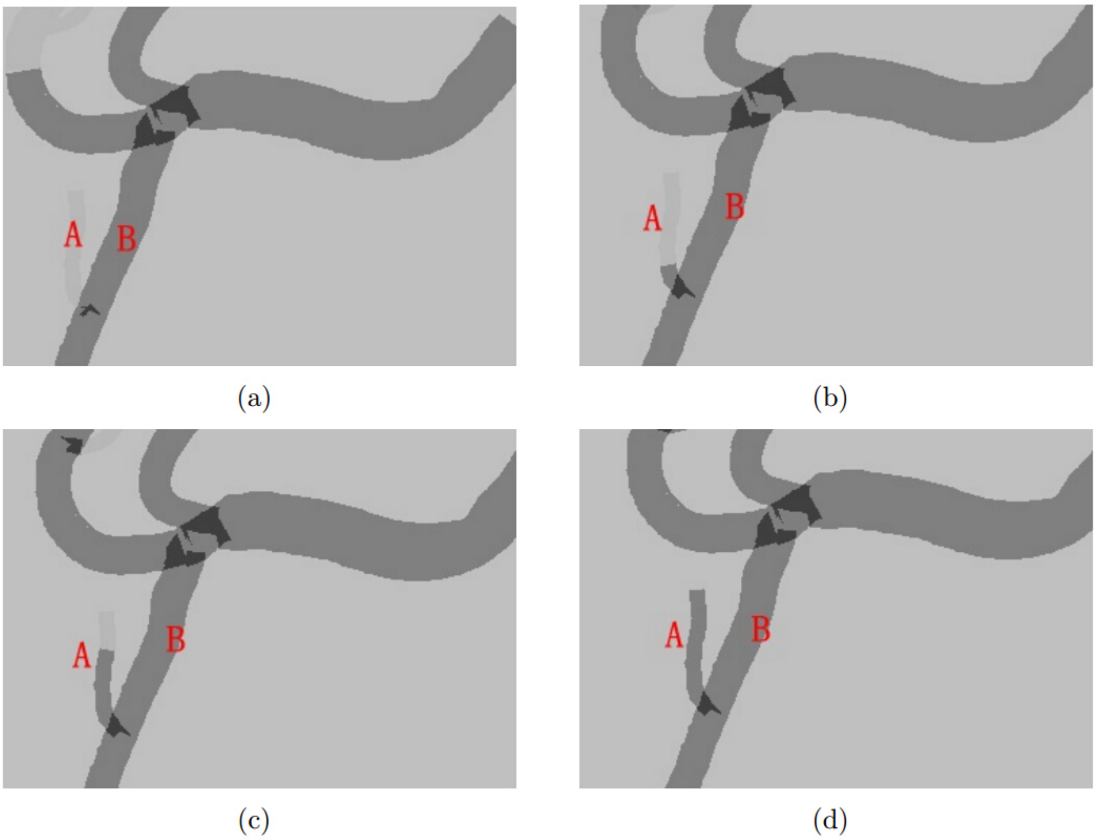


Fig. 6.9 Visualization of drug delivery in hepatic vessel tree of 4 discrete time steps.

than the results based on numerical simulations in literature [198], which varies from 13% to 22%. Nevertheless, as a result of the complexities of two-phase flows, errors of 30% are common in such simulations [199].

The results have demonstrated that the proposed method is efficient in providing robust and fast simulation of patient-specific drug delivery. It can also be used in evaluating the blood circulation for various types of geometries based on the same two-phase fluidic model. In the aspect of predicting flow trajectory in vascular system, this method is generally a simple but effective solution which achieves intuitive visual approximation of the simulated fluid. The method is also feasible to be implemented in contrast dye injection of angiography applications. Angiography is a medical imaging technique used to visualize the inside, or lumen, of blood vessels and organs of the body, with particular interest in the arteries, veins, and the heart chambers. The technique is traditionally done by injecting a radio-opaque contrast agent into the blood vessel and imaged using X-ray based techniques such as fluoroscopy. Since the proposed two-phase fluidic model is able to simulate the flow trajectory of the two fluids with different densities and viscosities, the surgical planning of angiography can also be predicted using a reconstructed patient-specific vascular model. Different from the conventional way of using generic drift-flux model to simulate the drug delivery, the geometrical model of the vascular system used in this work is patient-specific which allows preoperative planning of actual and upcoming patient cases based on the simulation results.

6.5 Summary

This chapter introduced a new method of simulating chemotherapy drug delivery in reconstructed patient-specific model of hepatic vascular system. The method used to describe the flow trajectory of drug fluid is based on a two-phased fluidic model which integrated both liquid fluid and air bubbles. A designed experiment is employed to perform bubble characterization and flow regime identification. According to the medical reports, the two-phase fluidic model more closely resembled the real drug delivery process since there is a significant amount of void fraction in natural human blood. Visualization of the flow trajectory has also been demonstrated using the proposed rendering technique. With further development and calibration of the fluidic model based on experiment and empirical data, this method can be used to assist on-site diagnostics and surgical planning of chemotherapy drug delivery.

Chapter 7

Simulation of Smoke During Endoscopic Vessel Harvesting

In this chapter, I introduce my investigation on physics-based surgical smoke simulation. Surgical smoke generated during the treatment could obscure clinician's field of view. In this work, a high-fidelity simulation of smoke using an improved vortex particle method based on a patient-specific artery model is proposed. The method is a combination of grid-based and particle-based method which avoids the computationally intensive meshing process in conventional grid-based method. The results have been demonstrated in an application of endoscopic vessel harvesting surgery, which could be used as a training scheme for junior clinicians. The method is also applicable to be used in other types of medical applications by adjusting the flow parameters of the smoke fluid.

7.1 Surgical smoke in operation theatre

Coronary artery bypass surgery is a treatment performed to relieve angina and reduce the risk of death from coronary artery disease. During this treatment, there is a very important minimally invasive surgical procedure called Endoscopic Vessel Harvesting (EVH). In this surgical procedure, a part of the vein from the patient's leg would be harvested and the proximal end of the vein attached to the aorta or one of its major branches [200]. A cautery tool is usually employed to divide the tissue of blood vessel with minimal blood loss. As the tissue of the patient's blood vessel is burnt, copious smoke would be generated. It is inevitable that clinicians and patients in operation theatre are exposed to the pollution of surgical smoke plume generated due to the thermal destruction of tissue.

Biologically, surgical smoke is from the decomposition of fat and protein by heat. Heat transferred from the tip of the cautery tool causes intracellular water to boil and cells to be ablated, and as a result, the tissue of the patient is burned. According to the medical report in [26], surgical smoke plume consists of about 95% water vapor and 5% of combustion by-products and cellular debris generated during the ablation process. In chemistry, these combustion by-products and cellular debris are regarded as a chemical and biological hazard which could cause various types of surgical risks.

It has been shown in in-vivo experiment that the mutagenic potency of condensates generated due to the burning of 1g biological tissue is equivalent to 6 unfiltered cigarettes [201]. Chemical study has revealed that there are many poisonous substances in surgical smoke such as formaldehyde, benzene, hydrocarbons, nitriles and hydrogen cyanide [202]. Moreover, infectious viral genes, viruses and viable cells have also been observed in surgical smoke plumes due to the non-combusted fraction of the biological tissue [203]. In addition, surgical smoke generated during EVH would significantly obscure the vision in the endoscope and could result in maloperations from the clinicians. Therefore, there is an increasing demand in developing realistic and physics-based simulation of smoke to assist surgical training and planning.

7.2 Methods

7.2.1 Model formulation

The design concept of the method is to generate high resolution results based on lower resolution simulations. The conventional grid-based method is able to simulate large-scale fluidic motion of smoke based on the Navier-Stokes (N-S) equations. However, the computational cost is very expensive due to the mesh generation at every time step. On the other hand, the conventional Vortex Particle Method (VPM) transforms the velocity form of the continuum equations into vorticity form by taking the curl of N-S equations. The particles advected in the simulation carry their own vorticity values, and the velocity field is reconstructed based on the vorticity computations of all particles in every time step. In fluid dynamics, VPM is regarded as a full Lagrangian method which has the advantage of restoring the lost details of grid-based method due to numerical dissipations. Nevertheless, the reconstruction of velocity field from vorticity is rather complicated, and hence, the graphic rendering of turbulent effects in smoke simulation is very computationally intensive.

The proposed method of smoke simulation is basically a combination of grid-based and particle-based method. The large scale flow motion of smoke is simulated using the grid-based method. The small details which are damped out in the grid-based method due to numerical dissipations are restored using the VPM. In order to reduce the computational cost, the grid-based velocity field is used to compute the vorticity of all the particles. Therefore, the initial velocity of each particle at the beginning of a time step is known, and the computation of vorticity equations would be easy to be solved.

Besides, a vorticity confinement force term is used to compute the lost details in grid-based smoke simulation instead of the conventional way of using first-order integration scheme. The lost details in grid-based method are normally the rolling and other turbulent effects. A vorticity forcing method is used to transfer the vorticity value of all particles to the grid at the end of time step. The effects of all particles are included in the vorticity confinement force and added to the continuum equations, and the velocity field in the next time step is then updated accordingly. Since the method avoids using arbitrarily assigned interventions to create the turbulent effects, which are common in conventional smoke simulations, the rendering of all the small details are physics-based.

7.2.2 Grid-based method

The surgical smoke is modeled as incompressible, single-phase, constant density and inviscid fluid. The reason for assuming smoke to be inviscid is to reduce the computational cost of solving N-S equations. In physics, this assumption is also valid because the influence of viscosity is negligible in gas-phase material [117]. The behavior of the incompressible fluid is described by the N-S equations

$$\frac{\partial u}{\partial t} = -(u \cdot \nabla)u - \frac{1}{\rho}\nabla p + \nu \nabla^2 u + f_e + f_{conf}, \tag{7.1}$$

$$\nabla \cdot u = 0, \tag{7.2}$$

where $u = (x_1, x_2, x_3)$ denotes the velocity, ρ is the density of smoke, p is the pressure, ν is the kinematic viscosity, f_e is the external force such as buoyancy and gravity, and f_{conf} is the vorticity confinement term used to transform particles' vorticities back to the velocity field. These two equations represent the conservation of mass and momentum in fluid dynamics. Because the smoke is assumed to be an inviscid fluid, the term $\nu \nabla^2 u$ vanishes.

The vorticity confinement force is used to model the small scale details of the smoke motion that are lost due to the coarse grid and numerical dissipation. In each time step, a weighted average interpolation of state values of the previous step is used to compute the new state values in the grid-based method. This interpolation operation could smooth out small details of the fluid motion which is called the numerical dissipation problem in grid-based methods. The vorticity confinement force is computed based on the vorticity of the fluid. The vorticity defines how the velocity vector changes when one moves by an infinitesimal distance in a direction perpendicular to it. Another way to understand the vorticity is to imagine that, a tiny part of the continuum becomes solid and the rest of the flow disappears at one particular moment. If this solid particle is rotating instead of just moving with the flow then there is vorticity in the flow.

In order to solve the N-S equations, the method described in [117] is used which divided the computation into two procedures. Equation 7.2 was first solved without the pressure term to obtain a temporary velocity field u' where

$$\frac{u'-u}{\Delta t} = -(u \cdot \nabla)u + f, \tag{7.3}$$

$$f = f_e + f_{conf}. \tag{7.4}$$

Since the smoke is assumed to be incompressible, the pressure term can be derived based on the Poisson's equation, where

$$\nabla^2 p = \frac{1}{\Delta t} \cdot u'. \tag{7.5}$$

In Neumann boundary condition, the change of pressure with respect to the normal of the boundary (n) equals to zero, which means $\frac{\partial p}{\partial n} = 0$. The temporary velocity was then made to be incompressible by subtracting the gradient of pressure from it, where

$$u = u' - \Delta t \nabla p. \tag{7.6}$$

The vorticity confinement force f_{conf} is used to restore the turbulent effects of smoke which is realized by transfering the small details computed by the VPM. The vorticity ω is obtained by taking the curl of velocity field, where

$$\omega = \nabla \times u. \tag{7.7}$$

The vorticity location vector N is then normalized by

$$N = \nabla|\omega|/|\nabla|\omega|| \tag{7.8}$$

The vector N basically points from the location of lower vorticity concentration to the location of higher vorticity concentration. The vorticity confinement force is then computed by

$$f_{conf} = \varepsilon h(N \times \omega), \tag{7.9}$$

where h is the size of the grid, and ε is a dimensionless constant used to calibrate the size of the small details added back to the original flow of the smoke fluid.

7.2.3 Improved vortex particle method

The first step of the vortex particle method is to transfer the velocity form of the N-S equations to vorticity equations by taking the curl of Equation (7.1). The first term on the left hand side (LHS) becomes $\partial\omega/\partial t$ since

$$\nabla \times \frac{\partial u}{\partial t} = \frac{\partial}{\partial t}(\nabla \times u) = \frac{\partial \omega}{\partial t}. \tag{7.10}$$

Expanding the first term on the right hand side (RHS) of Equation (7.1), we get

$$-(u \cdot \nabla)u = -\nabla(\frac{u^2}{2}) + u \times \omega. \tag{7.11}$$

Therefore, taking the curl of Equation (7.11), we will get

$$\nabla \times (-\nabla(\frac{u^2}{2}) + u \times \omega) = -\nabla \times \nabla(\frac{u^2}{2}) + \nabla \times (u \times \omega) = -(u \cdot \nabla)\omega + (\omega \cdot \nabla)u, \tag{7.12}$$

since

$$\nabla \times \nabla \cdot scalar = 0. \tag{7.13}$$

The second term on the RHS of Equation (7.1) also becomes zero because $\nabla \times \nabla \cdot scalar = 0$. The third and fourth terms on the RHS can be derived as

$$\nabla \times (\mu \nabla^2 u) = \mu \nabla \omega, \tag{7.14}$$

$$\nabla \times (f_e + f_{conf}) = \nabla \times f. \tag{7.15}$$

Hence, the vorticity form for the conservation of momentum in the N-S equations is given by

$$\frac{\partial \omega}{\partial t} = -(u \cdot \nabla)\omega + (\omega \cdot \nabla)u + \nu \nabla^2 \omega + \nabla \times f. \tag{7.16}$$

As it has been illustrated in the previous section, $\nu \nabla^2 \omega = 0$ for inviscid fluid. In conventional VPM, we need to solve the vorticity form of the N-S equations to obtain vorticity field and transform it back to reconstruct the velocity filed in the next time step. This procedure is usually implemented by the Biot-Savart Law which is very computationally intensive. For example, to compute the velocity of a vortex particle in two-dimensional space, we need to do the integration

$$u(x,t) = -\frac{1}{4\pi}\int_{\Omega}\frac{x-y}{||x-y||^3}\times\omega(y,t)dy, \tag{7.17}$$

where t is the time, Ω is the problem space, x and y are the corresponding quantities in the two axes.

In order to reduce the computational cost, the velocity field obtained from the grid-based method was used to compute the initial vorticities of the vortex particles at the beginning of each time step. This methodology was first introduced in [204] which has proven that such implementation would be computationally efficient without much trade-offs in accuracy.

In this work, the vortex particles were first randomly generated at the source. In EVH, this source was defined as the contact point of the cautery tool with the blood vessel. In each time step, a vortex particle p was advected based on the initial position and the associated velocity field, where

$$x_p(t+\Delta t) = x_p(t) + \Delta t \cdot u_p, \tag{7.18}$$

where x_p is the position of the vortex particle.

In most cases, the vortex particle would not be located exactly on the grid point. Hence, the velocity u_p was obtained from the interpolation of the neighboring 27 grid points. Therefore, the vortex advection term $(u\cdot\nabla)\omega$ in Equation (7.16) can be resolved. After that, the vortex stretching term $(\omega\cdot\nabla)u$ was evaluated by computing the derivatives of the velocity field at the location of the vortex particle where,

$$\omega_p(t+\Delta t) = \omega_p(t) + \Delta t(\omega_x\cdot\frac{\partial u}{\partial x} + \omega_y\cdot\frac{\partial u}{\partial y} + \omega_z\cdot\frac{\partial u}{\partial z}). \tag{7.19}$$

The update of vorticities of all the vortex particles was used to update the velocity field at the end of the time step. As it has been discussed previously, the implementation in this simulation was not based on the Biot-Savart Law which was computationally intensive. Instead, the vorticity forcing method described in [205] is used to influence the velocity field. The influence of each vortex particle on the grid map is defined by a clamped Gaussian kernel which is rotationally symmetric and decreasing with the distance of the particle's center. In fact, any kernel function with these properties could be used as the interpolation kernel, which is similar to the choice of smoothing kernel in SPH method discussed in Chapter 5. The clamped Gaussian kernel is defined by

$$W_p((x-x_p),r) = \begin{cases} \frac{e^{-|x-x_p|^2/2r^2}}{(r^3(2\pi)^{3/2})} & \text{if } 0 \leq |x-x_p| \leq r \\ 0 & otherwise. \end{cases} \tag{7.20}$$

where r is the control distance of the kernel. The interpolation kernel avoids computing the influence of one particle with respect to the whole grid map, which is very expensive in computational cost. Instead, it allows the program to only evaluate the influencing strength of the vortex particle within the control distance of the kernel. Those grids which are outside the control circle would not be affected by this vortex particle. Such a concept is demonstrated in Figure 7.1.

For a particular grid point a within the control distance, the vorticity influenced by the particle p is given by

$$\omega_a = W_p((x-x_p),r) \cdot \omega_p. \tag{7.21}$$

Hence, the vorticity confinement force used to update the velocity field can be derived as

$$F_a = |\omega_p||u \cdot \omega_a|(N_a \times \frac{\omega_a}{|\omega_a|}), \tag{7.22}$$

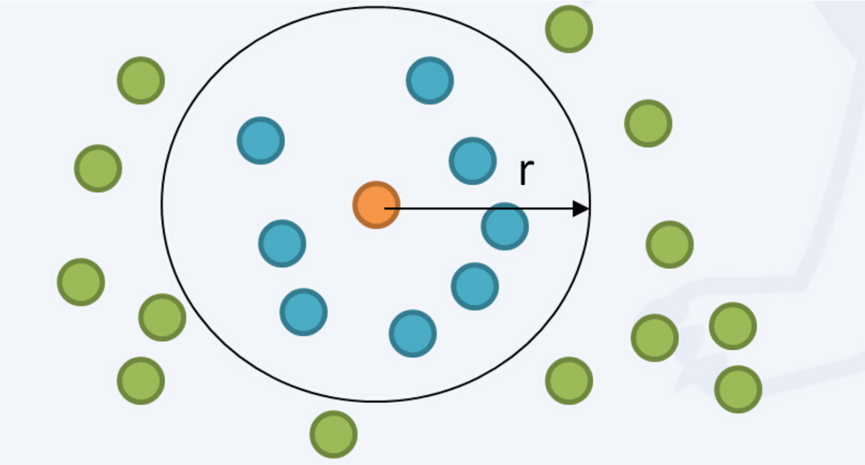


Fig. 7.1 2D representation of vorticity forcing method. Only the grid points within the control length r (blue color) would be influenced by the vortex particle (orange color). The grid points colored in green would not be affected.

where

$$N_a = (x_a - x_p)/|x_a - x_p|. \tag{7.23}$$

The magnitude of the influence can be controlled by multiplying a constant to F_a. It has been reported in [204] that if this constant is chosen to be too high, the simulation would become unstable. However, this choice is highly subjective and it can be calibrated based on the simulation results.

7.3 Results and Discussion

As it has been introduced in Section 7.1, the clinician needs to 'harvest' a segment of healthy blood vessel from another part of the patient's body, from the leg of the patient in this case, in EVH surgery. A cautery tool was used to divide the tissue of the blood vessel through a surgical incision. This work was collaborated with the National University Hospital (NUH) in Singapore to acquire the medical images taken during this surgical procedure. It can be seen in Figure 7.2 that the smoke generated during the surgery heavily affected the vision of the endoscope. Hence, a realistic and accurate simulation of smoke is important for the technical success of the surgical training and planning.

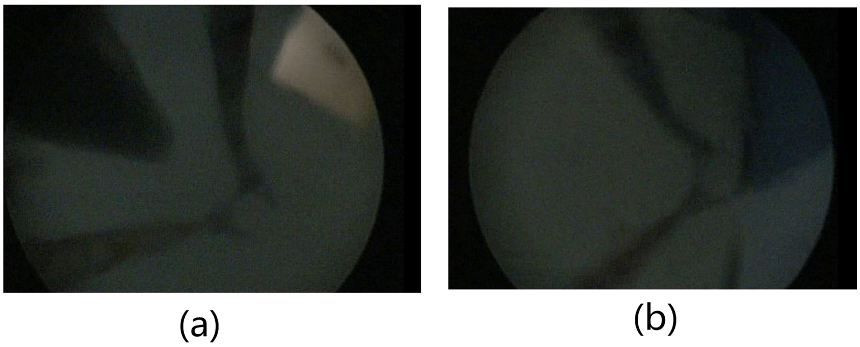


Fig. 7.2 Photos taken during the EVH surgery when smoke was generated

7.3.1 Smoke simulation

The computing platform used in the proposed simulation was an Intel(R) Core(TM)i5-3470 CPU @ 3.2GHz, 8G RAM with NVIDIA GeForce 605. In order to test the realism and interactivity of the simulation, a demo scenario has been constructed where a ball was dropped from the top of the smoke. The virtual ball was modeled as a rigid object with no external force applied on it except gravity. Solid boundary condition was implemented at the surface of the sphere to prevent the penetration of smoke particles. The computational domain is $1m \times 1m \times 1m$ divided by a $256 \times 256 \times 256$ grid map. The particles flow near the boundary of the computational domain would be deleted. The number of particles generated in this simulation is 1100. It can be seen in Figure 7.3 that the smoke flow simulated in this scenario resembled the rolling and stirring of smoke which were normally absent in the conventional simulations using coarse grid map. Besides, as shown in Figure 7.3 (c) and (d), the boundary condition is well defined and the smoke is able to interact with the solid object. Since the motion of the smoke was computed by vortex particles, the contact detections and the force interactions are easy to be done by using Newton's Second Law in solid mechanics.

In conventional grid-based method, such interactions are very computationally intensive, and realistic graphic rendering is usually absent from these simulations.

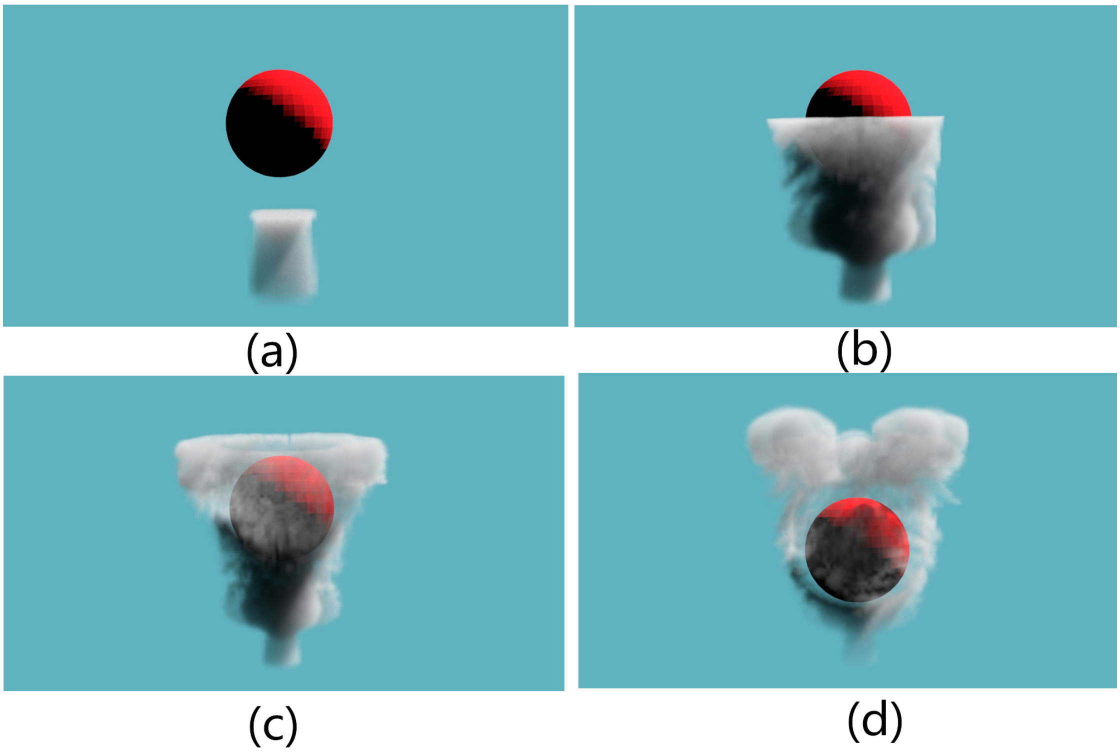


Fig. 7.3 Simulation of smoke interacting with solid object

7.3.2 Simulation of surgical smoke during EVH

A simulation of smoke generated by the cautery tool in EVH surgery is shown in Figure 7.4. The 3D models of blood vessel and the lumen in patient's leg were reconstructed from clinical CT-images. Hence, the simulation is patient-specific and can be used to assist diagnostics of the surgery. Based on the measurements in the operation theatre, clear smoke would be observed from the endoscope 7 to 11 seconds after the cautery tool started to ablate the tissue of blood vessel. Thus, the simulation defined the beginning time of the smoke generation to be 9 seconds after the cautery tool was in contact with the target segment of the blood vessel.

Technically, this parameter is arbitrarily assigned, and the users are able to customize the settings based on their specific applications.

The number of particles used in this simulation is 400, and the grid map is $128 \times 128 \times 384$. When the smoke was first generated in Figure 7.4(b), it grew rapidly from the source located on the contact point between the cautery tool and the blood vessel. This result converges with the fact the smoke fluid tends to move fast at the beginning when the temperature difference between the heat source and the ambient is large. The growing speed of the smoke can be controlled by adjusting the initial velocity of the vortex particles in the spatial domain. When the cautery tool stopped giving heat in Figure 7.4(c), there was still some heat generated until the temperature of the heat source became lower than the defined threshold. In actual EVH, most of the smoke would be gradually absorbed by the arterial wall due to the biological properties of human tissue, and Figure 7.4(d) shows the moment before the smoke vanished and was totally absorbed. The simulated endoscopic view of the smoke is shown in Figure 7.5. Compare with Figure 7.2, it can be seen that the simulated smoke generated during the surgery is a vital visual obstacle for the clinician. Due to the low resolution of endoscopy and the low light condition during EVH, patient-specific simulation of surgical smoke is important and has great potential in surgical training and planning.

7.3.3 Simulation of smoke in other medical applications

In addition to the surgical training and planning of EVH surgery, the proposed simulation of smoke can also be used in other types of medical applications involving ablation and burning of biological tissues. In fact, electrocautery ablation, laser ablation and harmonic scalpels will create different types of surgical smoke in operation theatre [25]. Figure 7.6 shows an application of the proposed smoke simulation in liver tissue ablation. The 3D liver model was also reconstructed from clinical CT-images. In this simulation, the initial velocity of the vortex particle was adjusted to a lower magnitude compare with the one used in EVH

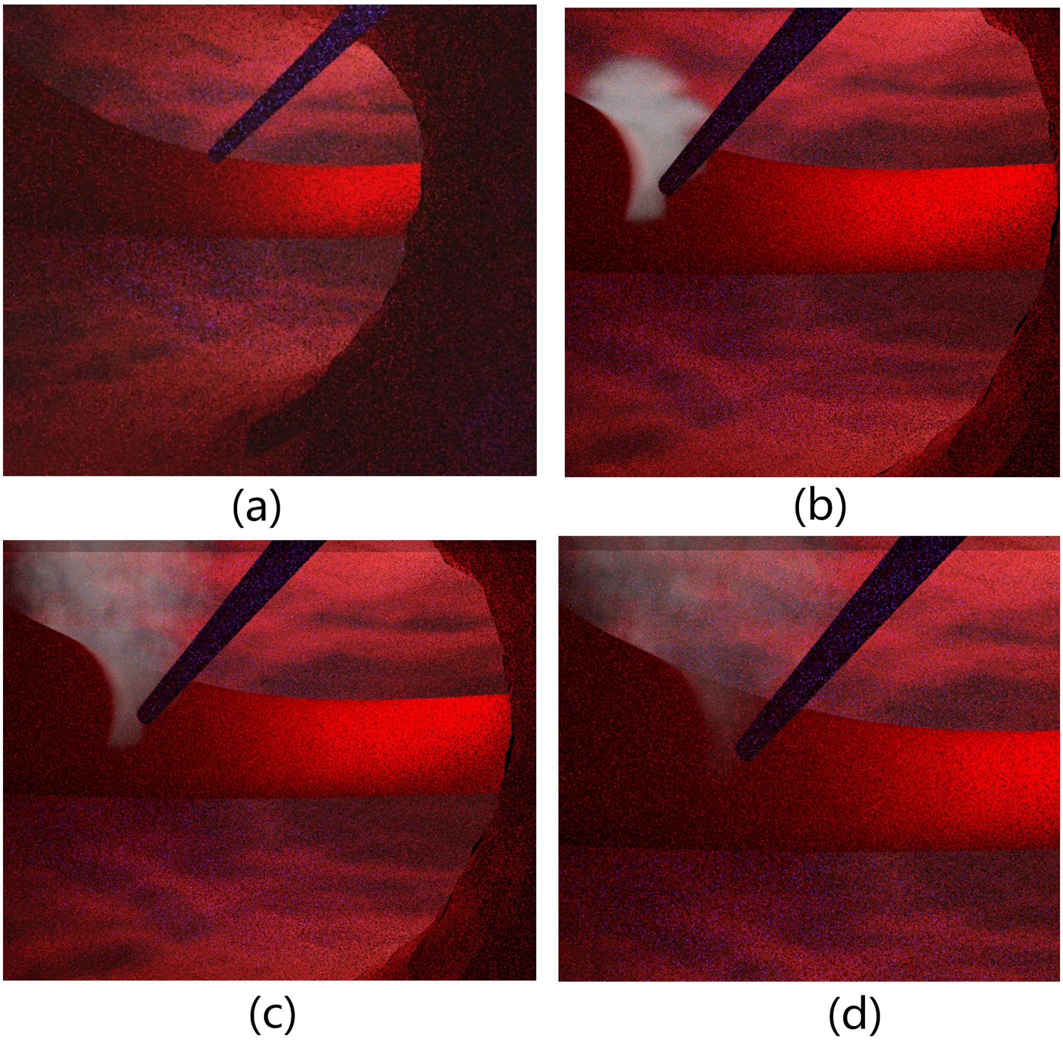


Fig. 7.4 Simulation of smoke during EVH surgery

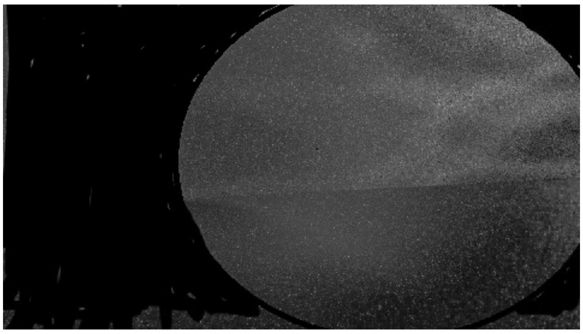


Fig. 7.5 Endoscopic view of smoke during EVH surgery

simulation. It can be seen that the smoke grew slower and a large portion of smoke would accumulated near the liver model. In actual liver tissue ablation surgery, the accumulation of smoke inside the patient's body is regarded as a medical hazard. External sucking device is usually used to evacuate the surgical smoke and to reduce its influence on the healthy tissue of the patient. The simulation resembled the accumulation of smoke in this surgical procedure. The method can be adapted in other types of medical applications by changing the corresponding fluidic parameters of the smoke.

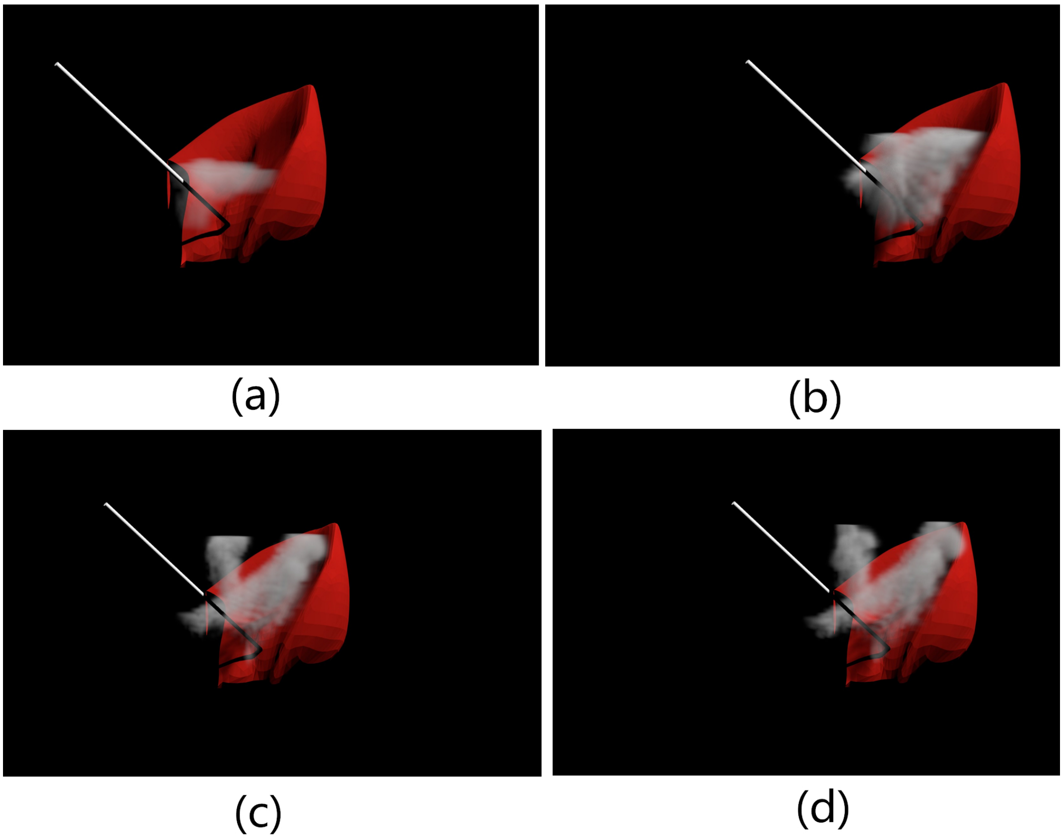


Fig. 7.6 Simulation of smoke in liver tissue ablation

7.4 Summary

The proposed method for smoke simulation combines the advantages of both grid-based and particle-based methods. The coarse mesh used in the grid-based method was able to predict the main flow trajectory of the smoke fluid. The resultant velocity field was used to compute the vorticity of particles in the vortex particle methods. Such an implementation reduced the intensive computation in conventional particle-based method. On the other side of the spectrum, the vorticity force obtained in the improved vortex particle method restored the lost details due to numerical dissipations in grid-based method. The simulation results were shown to be stable and realistic, and they can be used to assist the diagnostics in medical training and planning. High compatibility has also been shown in the result which allows the method to be used in other types of medical applications.

Chapter 8

Conclusions and Future Works

8.1 Conclusions

This dissertation focuses on the study of fast physics-based simulation of vascular surgery. In particular, this work has investigated three important aspects of physics phenomena in vascular surgery including the simulation of blood vessel deformation, hemodynamics, and surgical smoke. A component-based modeling framework for blood flow simulation has been proposed to integrate the soft body deformation component with the fluid component to produce realistic graphic rendering of patient-specific simulation. Surgical smoke generated due to the ablation of patient's tissue in operating theatre has also been studied using an improved vortex particle method. The results of these simulations have been shown to be high-fidelity, robust, and computationally efficient which can be used in various types of medical applications such as surgical training and planning.

The first investigation of this dissertation is the simulation of soft body deformation in vascular surgery. The work is based on an improved lumped element method to simulate the biomechanical properties of the soft tissue. Patient-specific models of blood vessel have been reconstructed from clinical CT-images to resemble the pathological structures of the patient. The method balances simulation realism and computational efficiency to allow the simulation to be performed in real time. GPU parallelization has been employed to accelerate the computation of the deformation matrix for the independent lumped elements. There is approximately 20 times speed up compare with that of CPU-only system. A novel approach of using entropy threshold to model the division of soft-bodied object has been proposed to simulate the tearing of blood vessel during surgery. Such an implementation avoids the intensive computation of the nonlinear plastic deformation before fracture in conventional FE-based methods. Graphic rendering of blood vessel deformation is shown to be realistic and computationally efficient, and the method can be used to model different types of soft-bodied objects.

The second investigation of this dissertation is the simulation of hemodynamics using meshfree method. Evaluation of fluid motion based on particles avoids the computationally intensive meshing process in conventional grid-based method. An improved SPH method has been proposed to simulate the blood flow of patients with computer graphic rendering. A quartic smoothing kernel function is used to reduce the computational cost by only evaluating the force interactions with neighboring particles within the defined smoothing length. The kernel is able to reduce the tensile instability compare with the conventional cubic splines. The proposed method has been performed in a simulation of injecting blood fluid into reconstructed patient's blood vessel model in 3-dimensional space. Important phenomena such as viscous flow, back flow, eddy forming/dissipation and free surface fluid have been achieved with high fidelity in the simulation.

Besides, an improved Finite Particle Method has been used to simulate the blood flow in heart and hand circulation with blockages. The method uses the integral form of function representation based on a transformed Taylor series expansion, and the parameters in the shape function are approximated by a set of basis functions. Dynamic medical images have been employed to predict the flow trajectory in the patient's vascular system. This is a new attempt of using meshfree method to evaluate the pressure distribution and velocity of blood flow in unhealthy human vascular. The outcomes of the blockage growth in the patient's blood vessel have been investigated using the proposed method. Such a simulation has the potential to be used as an alternative approach to assist the diagnosis of various types of vascular diseases.

In addition, a 2D simulation of chemotherapy drug delivery in patient-specific hepatic vascular system has been performed by using the two-phase fluidic model. In this work, human blood has been modeled as a mixture of liquid-phase plasma and air-phase bubbles. Bubble characterizations were obtained from a designed experiment of bubbly flow produced by injecting micro-bubbles into liquid-phase fluid. The rendering of two-phase fluid is based on the 'virtual tubelets' method where cylindrical billboard is used to represent the basis geometry of the fluid. Fast generation of flow trajectory has been demonstrated in the results. This simulation can be used in on-site diagnostics of drug delivery and surgical planning.

The third investigation of this dissertation is the simulation of surgical smoke during the ablation of human tissue. The flow motion of the smoke is simulated using an improved Vortex Particle Method. The method first evaluates the flow motion of smoke particles based on the coarse grid map to construct the initial velocity field, which is then used to compute the vorticities of vortex particles. Such an implementation could avoid the expensive computational cost in the conventional particle-based method of directly computing the vorticity based on the vorticity form of N-S equations. A vorticity forcing method is used to transform the influence of vortex particles to update the velocity field so that the lost turbulent effects could be restored. The rendering of surgical smoke is shown to be high realistic and computationally efficient in a simulation of endoscopic vessel harvesting surgery. Besides, the method can also be implemented in other surgical applications with different patient-specific models.

Physics-based simulation of vascular surgery is an active research area in medical engineering. It enables realistic surgical training and planning to be conducted in a risk-free manner. Investigations of physical phenomena are of great importance for developing a realistic and efficient surgical simulator. However, there are lack of explorations in the three aspects discussed in this dissertation. This work has contributed to the research and development of vascular surgery simulation.

8.2 Future works

8.2.1 Integration with haptic devices and augmented reality

Haptics is basically the force/touch feedback from external devices to create an augmented reality environment for the users. The works described in this dissertation have been focused on virtual reality simulations of vascular surgery without haptic feedback. There are generally two types of haptic feedback including kinesthetic (force) and cutaneous (tactile) feedbacks considered in medical applications. Augmented reality is able to better resemble the real surgery in trainings, and the users could learn the respective surgical skills by operating on the virtual models using real surgical tools with haptic guidance.

Investigations of augmented reality in medical applications have been focused on the development of surgical training systems based on specially design surgical tools or generic phantom devices. However, haptic feedback alone does not provide enough information to produce an immersive medical training simulation, invariably visual and sometimes auditory feedback are incorporated. The virtual models used in these applications are normally not physics-based or patient-specific. Our team is currently developing a training system which integrates the aorta model described in Chapter 4 with haptic surgical tools to train junior clinicians. The magnitude of force feedback is created based on the computation of aorta deformation. The system would learn the operational skills from masters and generate the haptic guidance to the trainees, and thus, it is able to obtain a high-fidelity and more effective surgical training scheme.

8.2.2 Advancement of computing techniques

In this dissertation, GPU parallelization has been used to compute the deformation matrix of meshed segment independently in soft body simulation. The implementation is shown to be efficient in this work. However, due to the advancement in hardware technology, this method does not make full use of the computing power of the new generation multi-core GPU and CPU. It has been reported that there are other frameworks of parallel computing which is able to combine the efficiency of CPU and GPU [128]. A method of assembly by nonzero entries using shared memory was recently proposed in [31, 60, 128]. The assembly of element data on shared memory yields a faster data update rate in real-time simulation. If such acceleration can be used in the surgical simulations, the accuracy of mass matrix and stiffness matrix could be further refined by using more complicated modeling method.

8.2.3 Experimental validation of hemodynamic simulation

The fluidic parameters of human blood used in this dissertation are based on generic data from medical reports. However, the density and viscosity of blood are highly personalized in different patients. Lack of experimental validation is perhaps the major deficiency of the proposed simulation. In order to further increase the accuracy, blood test could be employed to acquire the corresponding medical data from the patient so that it can produce a more realistic solution in hemodynamic simulation.

Moreover, an experiment of injecting a contrast agent into the patient's vascular system during the surgery could be performed to evaluate the accuracy of the simulation results. In fact, the injection of a contrast agent used in dynamic medical images has been utilized to predict the flow trajectory in the simulation of hand circulation described in Chapter 5. However, the validation of flow velocity will require more advanced and accurate experimental techniques to be considered in the future works.

References

[1] F. Saunders, E. Golden, R. D. White, and J. Rife, “Experimental verification of soft-robot gaits evolved using a lumped dynamic model,” *Robotica*, vol. 29, no. 06, pp. 823–830, 2011.

[2] A. A. Amsden and F. H. Harlow, “A simplified mac technique for incompressible fluid flow calculations,” *Journal of Computational Physics*, vol. 6, no. 2, pp. 322–325, 1970.

[3] S. McKee, M. Tomé, V. Ferreira, J. Cuminato, A. Castelo, F. Sousa, and N. Mangiavacchi, “The mac method,” *Computers & Fluids*, vol. 37, no. 8, pp. 907–930, 2008.

[4] J. U. Brackbill, D. B. Kothe, and H. M. Ruppel, “Flip: A low-dissipation, particle-in-cell method for fluid flow,” *Computer Physics Communications*, vol. 48, no. 1, pp. 25–38, 1988.

[5] I. Buck, “Gpu computing with nvidia cuda,” in *ACM SIGGRAPH*, vol. 7, 2007.

[6] A. Zorlak, A. Zorlak, A. Thomassen, O. Gerke, H. Munkholm, H. Mickley, and A. C. P. Diederichsen, “Patients with suspected coronary artery disease referred for examinations in the era of coronary computed tomography angiography,” *The American Journal of Cardiology*, vol. 116, no. 3, pp. 344 – 349, 2015.

[7] J. M. Lappé, J. L. Grodin, Y. Wu, C. Bott-Silverman, and L. Cho, “Prevalence and prediction of obstructive coronary artery disease in patients referred for valvular heart surgery,” *The American Journal of Cardiology*, vol. 116, no. 2, pp. 280 – 285, 2015.

[8] D. N. Ku, “Blood flow in artries,” *Annu. Rev. Fluid Mech*, vol. 29, p. 399–434, 1997.

[9] A. S. Go, D. Mozaffarian, V. L. Roger, E. J. Benjamin, and J. D. Berry, “Heart disease and stroke statistics—2014 update: A report from the american heart association,” *Circulation*, vol. 129, no. 3, pp. e28–e292, 2014.

[10] J. Lamelas and T. C. Nguyen, “Minimally invasive valve surgery: When less is more,” *Seminars in Thoracic and Cardiovascular Surgery*, vol. 27, no. 1, pp. 49 – 56, 2015.

[11] C.-L. Ang, B. P.-B. Tow, S. Fook, C.-M. Guo, J. L.-T. Chen, W.-M. Yue, and S.-B. Tan, “Minimally invasive compared with open lumbar laminotomy: no functional benefits at 6 or 24 months after surgery,” *The Spine Journal*, vol. 15, no. 8, pp. 1705 – 1712, 2015.

[12] R. Mirhashemi, B. L. Harlow, E. S. Ginsburg, L. B. Signorello, R. Berkowitz, and S. Feldman, "Predicting risk of complications with gynecologic laparoscopic surgery," *Predicting risk of complications with gynecologic laparoscopic surgery*, vol. 92, no. 2, pp. 327–331, 1998.

[13] I. Peterlik, M. Sedef, C. Basdogan, and L. Matyska, "Real-time visio-haptic interaction with static soft tissue models having geometric and material nonlinearity," *Computer and Graphics*, vol. 34, pp. 43–54, 2010.

[14] T. Djukic, V. Mandic, and N. Filipovic, "Virtual reality aided visulization of fluid flow simulations with application in medical education and diagnostics," *Computers in Biology and Medicine*, vol. 43, no. 12, pp. 2046–2052, 2013.

[15] V. G. JP, S. BR, and M. A, "Toward a model for local drug delivery in abdominal aortic aneurysms," *Ann N Y Acad Sc*, vol. 1085, pp. 396–399, 2006.

[16] D. Stefanidis, S. Arora, D. M. Parrack, G. G. Hamad, J. Capella, T. Grantcharov, D. R. Urbach, D. J. Scott, and D. B. Jones, "Research priorities in surgical simulation for the 21st century," *The American Journal of Surgery*, vol. 203, no. 1, pp. 49 – 53, 2012.

[17] S. C. Papaspyros, A. Kar, and D. O'Regan, "Surgical ergonomics. analysis of technical skills, simulation models and assessment methods," *International Journal of Surgery*, vol. 18, pp. 83 – 87, 2015.

[18] H.-H. Lin and L.-J. Lo, "Three-dimensional computer-assisted surgical simulation and intraoperative navigation in orthognathic surgery: A literature review," *Journal of the Formosan Medical Association*, vol. 114, no. 4, pp. 300 – 307, 2015.

[19] Z. Lian, C. K. Chui, and S. Teoh, "A biomechanical model for real-time simulation of pmma injection with haptics," *Computers in Biology and Medicine*, vol. 38, no. 3, pp. 304–312, 2008.

[20] P. Binh, C. Chui, S. Ong, and S. Chang, "Vascular flow rendering for interactive simulation of contrast and drugs injection," in *Proceedings of TENCON 2009*, IEEE Annual Conference Series, IEEE, IEEE, 2009.

[21] C. J. Mullany, "Coronary artery bypass surgery," *Circulation*, vol. 107, no. 3, pp. e21–e22, 2003.

[22] E. Cameron and W. Walker, "Coronary artery bypass surgery.," *BMJ*, vol. 300, no. 6734, pp. 1219–1220, 1990.

[23] B. W. Lytle and D. M. Cosgrove, "Coronary artery bypass surgery," *Current problems in surgery*, vol. 29, no. 10, pp. 743–807, 1992.

[24] W. L. Barrett and S. M. Garber, "Surgical smoke: a review of the literature," *Surgical endoscopy*, vol. 17, no. 6, pp. 979–987, 2003.

[25] D. Hill, J. O'Neill, R. Powell, and D. Oliver, "Surgical smoke–a health hazard in the operating theatre: A study to quantify exposure and a survey of the use of smoke extractor systems in uk plastic surgery units," *Journal of Plastic, Reconstructive & Aesthetic Surgery*, vol. 65, no. 7, pp. 911–916, 2012.

[26] W. S. Sawchuk, P. J. Weber, D. R. Lowy, and L. M. Dzubow, "Infectious papillomavirus in the vapor of warts treated with carbon dioxide laser or electrocoagulation: detection and protection," *Journal of the American Academy of Dermatology*, vol. 21, no. 1, pp. 41–49, 1989.

[27] X. Wu, M. S. Downes, T. Goktekin, and F. Tendick, "Adaptive nonlinear finite elements for deformable body simulation using dynamic progressive meshes," in *Computer Graphics Forum*, vol. 20, pp. 349–358, 2001.

[28] M. Matyka and M. Ollila, "Pressure model of soft body simulation," *Proceedings of Sigrad, UMEA*, pp. 29–34, 2003.

[29] S. Cotin, H. Delingette, and N. Ayache, "Real-time elastic deformations of soft tissues for surgery simulation," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 5, no. 1, pp. 62–73, 1999.

[30] L. Buatois, G. B. Caumon, and B. Levy, "A gpu implementation of a general sparse linear solver," *International Journal of Parallel Emergent and Distributed Systems*, vol. 23, no. 3, pp. 205–223, 2008.

[31] C. Cecka, A. Lew, and E. Darve, "Application of assembly of finite element methods on graphics processors for real-time elastodynamics," *GPU computing gem*, pp. 187–205, 2012.

[32] S. a. Glagov, C. Zarins, D. Giddens, and D. N. Ku, "Hemodynamics and atherosclerosis. insights and perspectives gained from studies of human arteries.," *Archives of pathology & laboratory medicine*, vol. 112, no. 10, pp. 1018–1031, 1988.

[33] W. R. Milnor, "Hemodynamics," 1982.

[34] J. D. Anderson and J. Wendt, *Computational fluid dynamics*, vol. 206. Springer, 1995.

[35] G. K. Batchelor, *An introduction to fluid dynamics*. Cambridge university press, 2000.

[36] P. J. Roache, *Computational fluid dynamics*. Hermosa publishers, 1972.

[37] N. Rasmussen, D. Q. Nguyen, W. Geiger, and R. Fedkiw, "Smoke simulation for large scale phenomena," in *ACM Transactions on Graphics (TOG)*, vol. 22, pp. 703–707, ACM, 2003.

[38] A. Selle, A. Mohr, and S. Chenney, "Cartoon rendering of smoke animations," in *Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering*, pp. 57–60, ACM, 2004.

[39] A. Treuille, A. McNamara, Z. Popović, and J. Stam, "Keyframe control of smoke simulations," *ACM Transactions on Graphics (TOG)*, vol. 22, no. 3, pp. 716–723, 2003.

[40] D. L. Westphal and O. B. Toon, "Simulations of microphysical, radiative, and dynamical processes in a continental-scale forest fire smoke plume," *Journal of Geophysical Research: Atmospheres (1984–2012)*, vol. 96, no. D12, pp. 22379–22400, 1991.

[41] R. Fedkiw, J. Stam, and H. W. Jensen, "Visual simulation of smoke," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pp. 15–22, ACM, 2001.

[42] J. R. Levick, *An introduction to cardiovascular physiology*. Butterworth-Heinemann, 2013.

[43] R. Dampney, "Functional organization of central pathways regulating the cardiovascular system," *Physiological reviews*, vol. 74, no. 2, pp. 323–364, 1994.

[44] L. H. Opie, *Heart physiology: from cell to circulation*. Lippincott Williams & Wilkins, 2004.

[45] J. E. Hall, *Guyton and Hall textbook of medical physiology*. Elsevier Health Sciences, 2010.

[46] K. M. Anderson, P. M. Odell, P. W. Wilson, and W. B. Kannel, "Cardiovascular disease risk profiles," *American heart journal*, vol. 121, no. 1, pp. 293–298, 1991.

[47] H. S. Collaboration *et al.*, "Homocysteine and risk of ischemic heart disease and stroke: a meta-analysis," *Jama*, vol. 288, no. 16, pp. 2015–2022, 2002.

[48] J. A. Diamond and R. A. Phillips, "Hypertensive heart disease," *Hypertension research*, vol. 28, no. 3, pp. 191–202, 2005.

[49] E. V. Allen, N. W. Barker, and E. A. Hines, "Peripheral vascular diseases.," *Academic Medicine*, vol. 21, no. 5, p. 319, 1946.

[50] E. Marijon, M. Mirabel, D. S. Celermajer, and X. Jouven, "Rheumatic heart disease," *The Lancet*, vol. 379, no. 9819, pp. 953–964, 2012.

[51] M. Carbajo, J. M. Del Olmo, J. Blanco, C. De la Cuesta, M. Toledano, F. Martin, C. Vaquero, and L. Inglada, "Laparoscopic treatment vs open surgery in the solution of major incisional and abdominal wall hernias with mesh," *Surgical endoscopy*, vol. 13, no. 3, pp. 250–252, 1999.

[52] J. Teran, S. Blemker, V. Hing, and R. Fedkiw, "Finite volume methods for the simulation of skeletal muscle," in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 68–74, Eurographics Association, 2003.

[53] M. Nesme, F. Faure, and Y. Payan, "Hierarchical multi-resolution finite element model for soft body simulation," in *Biomedical Simulation*, pp. 40–47, Springer, 2006.

[54] J.-P. Gourret, N. M. Thalmann, and D. Thalmann, "Simulation of object and human skin formations in a grasping task," in *ACM Siggraph Computer Graphics*, vol. 23, pp. 21–30, ACM, 1989.

[55] G. Debunne, M. Desbrun, M.-P. Cani, and A. H. Barr, "Dynamic real-time deformations using space & time adaptive sampling," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pp. 31–36, ACM, 2001.

[56] S. F. Gibson and B. Mirtich, "A survey of deformable modeling in computer graphics," tech. rep., Citeseer, 1997.

[57] A. Gregory, A. Mascarenhas, S. Ehmann, M. Lin, and D. Manocha, “Six degree-of-freedom haptic display of polygonal models,” in *Proceedings of the conference on Visualization’00*, pp. 139–146, IEEE Computer Society Press, 2000.

[58] J. Georgii and R. Westermann, “A multigrid framework for real-time simulation of deformable bodies,” *Computers & Graphics*, vol. 30, no. 3, pp. 408–415, 2006.

[59] M. Shiraishi, H. Yoshinaga, A. Miyori, and E. Takahashi, “Simulation of dynamically rolling tire,” *Tire Science and Technology*, vol. 28, no. 4, pp. 264–276, 2000.

[60] J. Allard, H. Courtecuisse, and F. Faure, “Implicit fem solver on gpu for interactive deformation simulation,” *GPU Computing Gems Jade Edition*, pp. 281–294, 2011.

[61] G. Debunne, M. Desbrun, A. Barr, and M.-P. Cani, *Interactive multiresolution animation of deformable models*. Springer, 1999.

[62] G. Debunne, M. Desbrun, M.-P. Cani, and A. H. Barr, “Adaptive simulation of soft bodies in real-time,” in *Computer Animation 2000*, pp. 133–144, 2000.

[63] A. Pentland and J. Williams, *Good vibrations: Modal dynamics for graphics and animation*, vol. 23. ACM, 1989.

[64] P. Faloutsos, M. Van De Panne, and D. Terzopoulos, “Dynamic free-form deformations for animation synthesis,” *Visualization and Computer Graphics, IEEE Transactions on*, vol. 3, no. 3, pp. 201–214, 1997.

[65] M. Müller and M. Gross, “Interactive virtual materials,” in *Proceedings of Graphics Interface 2004*, pp. 239–246, Canadian Human-Computer Communications Society, 2004.

[66] M. Nesme, P. G. Kry, L. Jeřábková, and F. Faure, “Preserving topology and elasticity for embedded deformable models,” *ACM Transactions on Graphics (TOG)*, vol. 28, no. 3, p. 52, 2009.

[67] A. Maciel, T. Halic, Z. Lu, L. P. Nedel, and S. De, “Using the physx engine for physics-based virtual surgery with force feedback,” *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 5, no. 3, pp. 341–353, 2009.

[68] M. G. Choi and H.-S. Ko, “Modal warping: Real-time simulation of large rotational deformation and manipulation,” *Visualization and Computer Graphics, IEEE Transactions on*, vol. 11, no. 1, pp. 91–101, 2005.

[69] A. Nealen, M. Müller, R. Keiser, E. Boxerman, and M. Carlson, “Physically based deformable models in computer graphics,” in *Computer Graphics Forum*, vol. 25, pp. 809–836, Wiley Online Library, 2006.

[70] E. Sifakis and J. Barbic, “Fem simulation of 3d deformable solids: a practitioner’s guide to theory, discretization and model reduction,” in *ACM SIGGRAPH 2012 Courses*, p. 20, ACM, 2012.

[71] S. Cotin, H. Delingette, and N. Ayache, “A hybrid elastic model for real-time cutting, deformations, and force feedback for surgery training and simulation,” *The Visual Computer*, vol. 16, no. 8, pp. 437–452, 2000.

[72] H. Delingette, “Toward realistic soft-tissue modeling in medical simulation,” *Proceedings of the IEEE*, vol. 86, no. 3, pp. 512–523, 1998.

[73] T. Yang, C. K. Chui, R. Q. Yu, J. Qin, and S. K. Chang, “Quasi-linear viscoelastic modeling of arterial wall for surgical simulation,” *International journal of computer assisted radiology and surgery*, vol. 6, no. 6, pp. 829–838, 2011.

[74] D. Steinman, D. Vorp, and C. Ethier, “Computational modeling of arterial biomechanics: insights into pathogenesis and treatment of vascular disease,” *Journal of Vascular Surgery*, vol. 37, no. 5, pp. 1118–1128, 2003.

[75] Y. Zhong, B. Shirinzadeh, J. Smith, and C. Gu, “Soft tissue deformation with reaction-diffusion process for surgery simulation,” *Journal of Visual Languages & Computing*, vol. 23, no. 1, pp. 1–12, 2012.

[76] S. Muraki and Y. Kita, “A survey of medical applications of 3d image analysis and computer graphics,” *Systems and Computers in Japan*, vol. 37, no. 1, pp. 13–46, 2006.

[77] M. Sermesant, H. Delingette, and N. Ayache, “An electromechanical model of the heart for image analysis and simulation,” *Medical Imaging, IEEE Transactions on*, vol. 25, no. 5, pp. 612–625, 2006.

[78] Y. Zhong, B. Shirinzadeh, G. Alici, and J. Smith, “A reaction-diffusion methodology for soft object simulation,” in *Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*, pp. 213–220, ACM, 2006.

[79] R. Bridson, *Fluid simulation for computer graphics*. CRC Press, 2008.

[80] J. Stam, “Stable fluids,” in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pp. 121–128, ACM Press/Addison-Wesley Publishing Co., 1999.

[81] F. H. Harlow, “The particle-in-cell method for numerical solution of problems in fluid dynamics,” tech. rep., Los Alamos Scientific Lab., N. Mex., 1962.

[82] M. W. Evans, F. H. Harlow, and E. Bromberg, “The particle-in-cell method for hydrodynamic calculations,” tech. rep., DTIC Document, 1957.

[83] S. Koshizuka and Y. Oka, “Moving-particle semi-implicit method for fragmentation of incompressible fluid,” *Nuclear science and engineering*, vol. 123, no. 3, pp. 421–434, 1996.

[84] A. Gilmanov and S. Acharya, “A hybrid immersed boundary and material point method for simulating 3d fluid–structure interaction problems,” *International journal for numerical methods in fluids*, vol. 56, no. 12, pp. 2151–2177, 2008.

[85] A. R. York, D. Sulsky, and H. L. Schreyer, “Fluid–membrane interaction based on the material point method,” *International Journal for Numerical Methods in Engineering*, vol. 48, no. 6, pp. 901–924, 2000.

[86] M. Liu and G. Liu, “Smoothed particle hydrodynamics (sph): an overview and recent developments,” *Arch Comput Methods Eng*, vol. 17, pp. 25–76, 2010.

[87] J. D. Anderson, *Computational fluid dynamics: the basics with applications*. McGraw Hill, New York, 2002.

[88] F. H. Harlow, J. E. Welch, *et al.*, "Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface," *Physics of fluids*, vol. 8, no. 12, p. 2182, 1965.

[89] M. O. Deville, "Numerical experiments on the mac code for a slow flow," *Journal of Computational Physics*, vol. 15, no. 3, pp. 362–374, 1974.

[90] W. E. Pracht, "A numerical method for calculating transient creep flows," *Journal of Computational Physics*, vol. 7, no. 1, pp. 46–60, 1971.

[91] M. Golafshani, "A simple numerical technique for transient creep flows with free surfaces," *International journal for numerical methods in fluids*, vol. 8, no. 8, pp. 897–912, 1988.

[92] A. Fournier and W. T. Reeves, "A simple model of ocean waves," in *ACM Siggraph Computer Graphics*, vol. 20, pp. 75–84, ACM, 1986.

[93] J. Yu and G. Turk, "Reconstructing surfaces of particle-based fluids using anisotropic kernels," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 1, p. 5, 2013.

[94] D. Cline, D. Cardon, and P. K. Egbert, "Fluid flow for the rest of us: Tutorial of the marker and cell method in computer graphics," 2013.

[95] D. Sulsky, Z. Chen, and H. L. Schreyer, "A particle method for history-dependent materials," *Computer methods in applied mechanics and engineering*, vol. 118, no. 1, pp. 179–196, 1994.

[96] R. Ando and R. Tsuruno, "High-frequency aware pic/flip in liquid animation," in *ACM SIGGRAPH ASIA 2010 Sketches*, p. 25, ACM, 2010.

[97] Y. Zhu and R. Bridson, "Animating sand as a fluid," *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3, pp. 965–972, 2005.

[98] L. Lucy, "A numerical approach to the testing of the fission hypothesis," *Astron Journal*, vol. 82, no. 12, pp. 1013–1024, 1977.

[99] R. A. Gingold and J. J. Monaghan, "Smoothed particle hydrodynamics—theory and application to non-spherical stars," *Mon Not R Astron Society*, vol. 181, pp. 375–389, 1977.

[100] J. J. Monaghan, "Simulating free surface flows with sph," *Journal of computational physics*, vol. 110, no. 2, pp. 399–406, 1994.

[101] R. K. Noutcheuwa and R. G. Owens, "A new incompressible smoothed particle hydrodynamics—immersed boundary method," *Int. J. Numer. Anal. Model*, vol. 3, no. 2, pp. 126–167, 2012.

[102] M. Becker and M. Teschner, "Weakly compressible sph for free surface flows," in *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 209–217, Eurographics Association, 2007.

[103] M. Müller, D. Charypar, and M. Gross, “Particle-based fluid simulation for interactive applications,” in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 154–159, Eurographics Association, 2003.

[104] A. Vázquez-Quesada and M. Ellero, “Sph simulations of a viscoelastic flow around a periodic array of cylinders confined in a channel,” *Journal of Non-Newtonian Fluid Mechanics*, vol. 167, pp. 1–8, 2012.

[105] M. Hashemi, R. Fatehi, and M. Manzari, “A modified sph method for simulating motion of rigid bodies in newtonian fluid flows,” *International Journal of Non-Linear Mechanics*, vol. 47, no. 6, pp. 626–638, 2012.

[106] J. Wu, C. K. Chui, P. Binh, and C. L. Teo, “Real-time rendering of drug injection and interactive simulation of vessel deformation using gpu,” in *Engineering in Medicine and Biology Society (EMBC)*, 2013 35th Annual International Conference of the IEEE, IEEE, IEEE Press, 2013.

[107] C. Sheng, S. Sarwal, K. Watts, and A. Marble, “Computational simulation of blood flow in human systemic circulation incorporating an external force field,” *Medical and Biological Engineering and Computing*, vol. 33, no. 1, pp. 8–17, 1995.

[108] A. Quarteroni, “What mathematics can do for the simulation of blood circulation,” *MOX Report*, 2006.

[109] A. Pries, T. Secomb, P. Gaehtgens, and J. Gross, “Blood flow in microvascular networks. experiments and simulation.,” *Circulation research*, vol. 67, no. 4, pp. 826–834, 1990.

[110] Y. Tokuda, M.-H. Song, Y. Ueda, A. Usui, T. Akita, S. Yoneyama, and S. Maruyama, “Three-dimensional numerical simulation of blood flow in the aortic arch during cardiopulmonary bypass,” *European Journal of Cardio-thoracic Surgery*, vol. 33, no. 2, pp. 164–167, 2008.

[111] J. Jung, R. W. Lyczkowski, C. B. Panchal, and A. Hassanein, “Multiphase hemodynamic simulation of pulsatile flow in a coronary artery,” *Journal of biomechanics*, vol. 39, no. 11, pp. 2064–2073, 2006.

[112] A. Chaniotis, L. Kaiktsis, D. Katritsis, E. Efstathopoulos, I. Pantos, and V. Marmarellis, “Computational study of pulsatile blood flow in prototype vessel geometries of coronary segments,” *Physica Medica*, vol. 26, no. 3, pp. 140–156, 2010.

[113] C. W. Hirt and B. D. Nichols, “Volume of fluid (vof) method for the dynamics of free boundaries,” *Journal of computational physics*, vol. 39, no. 1, pp. 201–225, 1981.

[114] J. A. Sethian, *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, vol. 3. Cambridge university press, 1999.

[115] K. B. McGrattan, H. R. Baum, and R. G. Rehm, “Numerical simulation of smoke plumes from large oil fires,” *Atmospheric Environment*, vol. 30, no. 24, pp. 4125–4136, 1996.

[116] H. Baum, K. McGrattan, and R. Rehm, “Simulation of smoke plumes from large pool fires,” in *Symposium (International) on Combustion*, vol. 25, pp. 1463–1469, Elsevier, 1994.

[117] R. Fedkiw, J. Stam, and H. W. Jensen, “Visual simulation of smoke,” in *Proceedings of SIGGRAPH 2001* (E. Fiume, ed.), Computer Graphics Proceedings, Annual Conference Series, pp. 15–22, ACM, ACM Press / ACM SIGGRAPH, 2001.

[118] G. Y. Gardner, “Visual simulation of clouds,” in *ACM Siggraph Computer Graphics*, vol. 19, pp. 297–304, ACM, 1985.

[119] D. S. Ebert and R. E. Parent, “Rendering and animation of gaseous phenomena by combining fast volume and scanline a-buffer techniques,” in *ACM SIGGRAPH Computer Graphics*, vol. 24, pp. 357–366, ACM, 1990.

[120] J. Stam and E. Fiume, “Turbulent wind fields for gaseous phenomena,” in *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pp. 369–376, ACM, 1993.

[121] J. T. Kajiya and B. P. Von Herzen, “Ray tracing volume densities,” in *ACM SIGGRAPH Computer Graphics*, vol. 18, pp. 165–174, ACM, 1984.

[122] B. Engquist and A. Majda, “Absorbing boundary conditions for numerical simulation of waves,” *Proceedings of the National Academy of Sciences*, vol. 74, no. 5, pp. 1765–1766, 1977.

[123] J. Eldredge, T. Colonius, and A. Leonard, “A vortex particle method for compressible flows,” *AIAA Pap*, vol. 2641, pp. 1–9, 2001.

[124] N. Foster and R. Fedkiw, “Practical animation of liquids,” in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pp. 23–30, ACM, 2001.

[125] J. Starn, “A simple fluid solver based on the fft,” *Journal of graphics tools*, vol. 6, no. 2, pp. 43–52, 2001.

[126] G. D. Yngve, J. F. O’Brien, and J. K. Hodgins, “Animating explosions,” in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 29–36, ACM Press/Addison-Wesley Publishing Co., 2000.

[127] M. Garland, “Parallel computing with cuda,” in *Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium on*, pp. 1–1, IEEE, 2010.

[128] J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips, “Gpu computing,” *Proceedings of the IEEE*, vol. 96, no. 5, pp. 879–899, 2008.

[129] J. Nickolls and W. J. Dally, “The gpu computing era,” *IEEE micro*, no. 2, pp. 56–69, 2010.

[130] J. Fung and S. Mann, “Using multiple graphics cards as a general purpose parallel computer: Applications to computer vision,” in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 1, pp. 805–808, IEEE, 2004.

[131] J.-P. Farrugia, P. Horain, E. Guehenneux, and Y. Alusse, “Gpucv: A framework for image processing acceleration with graphics processors,” in *2006 IEEE International Conference on Multimedia and Expo*, pp. 585–588, IEEE, 2006.

[132] E. Olmedo, J. De La Calleja, A. Benitez, M. A. Medina, and L. d. P. por Computadora, “Point to point processing of digital images using parallel computing,” *International Journal of Computer Science Issues*, vol. 9, no. 33, p. 3, 2012.

[133] H. Courtecuisse, H. Jung, J. Allard, C. Duriez, D. Y. Lee, and S. Cotin, “Gpu-based real-time soft tissue deformation with cutting and haptic feedback,” *Progress in biophysics and molecular biology*, vol. 103, no. 2, pp. 159–168, 2010.

[134] Z. Taylor, M. Cheng, S. Ourselin, *et al.*, “High-speed nonlinear finite element analysis for surgical simulation using graphics processing units,” *Medical Imaging, IEEE Transactions on*, vol. 27, no. 5, pp. 650–663, 2008.

[135] G. R. Joldes, A. Wittek, and K. Miller, “Real-time nonlinear finite element computations on gpu–application to neurosurgical simulation,” *Computer methods in applied mechanics and engineering*, vol. 199, no. 49, pp. 3305–3314, 2010.

[136] K. Miller, A. Wittek, G. Joldes, A. Horton, T. Dutta-Roy, J. Berger, and L. Morriss, “Modelling brain deformations for computer-integrated neurosurgery,” *International Journal for Numerical Methods in Biomedical Engineering*, vol. 26, no. 1, pp. 117–138, 2010.

[137] A. Wittek, G. Joldes, M. Couton, S. K. Warfield, and K. Miller, “Patient-specific non-linear finite element modelling for predicting soft organ deformation in real-time; application to non-rigid neuroimage registration,” *Progress in biophysics and molecular biology*, vol. 103, no. 2, pp. 292–303, 2010.

[138] R. Wen, L. Yang, C.-K. Chui, K.-B. Lim, and S. Chang, “Intraoperative visual guidance and control interface for augmented reality robotic surgery,” in *Control and Automation (ICCA), 2010 8th IEEE International Conference on*, pp. 947–952, IEEE, 2010.

[139] E. Coumans, “Bullet physics engine,” *Open Source Software: http://bulletphysics. org*, vol. 1, 2010.

[140] G. Echeverria, N. Lassabe, A. Degroote, and S. Lemaignan, “Modular open robots simulation engine: Morse,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 46–51, IEEE, 2011.

[141] S. Marks, J. Windsor, and B. Wünsche, “Evaluation of game engines for simulated surgical training,” in *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*, pp. 273–280, ACM, 2007.

[142] J. R. Juang, W. H. Hung, and S. C. Kang, “Using game engines for physical–based simulations–a forklift,” 2011.

[143] A. Comsol, “Comsol multiphysics user’s guide,” *Version: September*, 2005.

[144] Hibbitt, Karlsson, and Sorensen, *ABAQUS/standard user's Manual*, vol. 1. Hibbitt, Karlsson & Sorensen, 2001.

[145] G. Van Rossum *et al.*, "Python programming language.," in *USENIX Annual Technical Conference*, vol. 41, 2007.

[146] E. Anton, "Principles of object-oriented software development," *Anton Eli ns., Harlow: Addison-Wesley*, vol. 5, p. 17, 2000.

[147] K. McInnis, "Component-based development. the concepts, technology and methodology," *CASTEK, http://www. cbd-hq. com/PDFs/cbdhq_000901km_cbd_con_tech_method. pdf*, 2000.

[148] M. Ishii and T. Hibiki, "Drift-flux model," in *Thermo-Fluid Dynamics of Two-Phase Flow*, pp. 361–395, Springer, 2011.

[149] T. Hibiki and M. Ishii, "One-dimensional drift-flux model and constitutive equations for relative motion between phases in various two-phase flow regimes," *International Journal of Heat and Mass Transfer*, vol. 46, no. 25, pp. 4935–4948, 2003.

[150] M. B. Liu and G. R. Liu, "Smoothed particle hydrodynamics (sph): an overview and recent developments," *Computational Methods in Engineering*, vol. 17, no. 1, pp. 25–76, 2010.

[151] A. Hérault, G. Bilotta, and R. A. Dalrymple, "Sph on gpu with cuda," *Journal of Hydraulic Research*, vol. 48, no. S1, pp. 74–79, 2010.

[152] S. Green, "Particle simulation using cuda," *NVIDIA whitepaper*, 2010.

[153] S. Haack, S. K. Nielsen, J. C. Lindegaard, J. Gelineck, and K. Tanderup, "Applicator reconstruction in mri 3d image-based dose planning of brachytherapy for cervical cancer," *Radiotherapy and Oncology*, vol. 91, no. 2, pp. 187–193, 2009.

[154] M. A. McAteer, J. E. Schneider, K. Clarke, S. Neubauer, K. M. Channon, and R. P. Choudhury, "Quantification and 3d reconstruction of atherosclerotic plaque components in apolipoprotein e knockout mice using ex vivo high-resolution mri," *Arteriosclerosis, thrombosis, and vascular biology*, vol. 24, no. 12, pp. 2384–2390, 2004.

[155] E. Catmull and J. Clark, "Recursively generated b-spline surfaces on arbitrary topological meshes," *Computer-aided design*, vol. 10, no. 6, pp. 350–355, 1978.

[156] C. M. Ma and M. Sonka, "A fully parallel 3d thinning algorithm and its applications," *Computer vision and image understanding*, vol. 64, no. 3, pp. 420–433, 1996.

[157] T. Wang and I. Cheng, "Generation of unit-width curve skeletons based on valence driven spatial median (vdsm)," in *Advances in Visual Computing*, pp. 1051–1060, Springer, 2008.

[158] S. E. Benzley, E. Perry, K. Merkley, B. Clark, and G. Sjaardama, "A comparison of all hexagonal and all tetrahedral finite element meshes for elastic and elasto-plastic analysis," in *Proceedings, 4th International Meshing Roundtable*, vol. 17, pp. 179–191, Sandia National Laboratories Albuquerque, NM, 1995.

[159] J. Barbič, F. Sin, and E. Grinspun, "Interactive editing of deformable simulations," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, p. 70, 2012.

[160] J. Barbic, *Real-time reduced large-deformation models and distributed contact for computer graphics and haptics*. PhD thesis, Intel, 2007.

[161] J. Barbič and D. L. James, "Real-time subspace integration for st. venant-kirchhoff deformable models," in *ACM transactions on graphics (TOG)*, vol. 24, pp. 982–990, ACM, 2005.

[162] F. J. Carter, T. G. Frank, P. J. Davies, D. McLean, and A. Cuschieri, "Measurements and modelling of the compliance of human and porcine organs," *Medical Image Analysis*, vol. 5, no. 4, pp. 231–236, 2001.

[163] M. Liu and D. Gorman, "Formulation of rayleigh damping and its extensions," *Computers & structures*, vol. 57, no. 2, pp. 277–285, 1995.

[164] M. D. McGarry and E. E. Van Houten, "Use of a rayleigh damping model in elastography," *Medical & biological engineering & computing*, vol. 46, no. 8, pp. 759–766, 2008.

[165] M. A. Crisfield, "A consistent co-rotational formulation for non-linear, three-dimensional, beam-elements," *Computer methods in applied mechanics and engineering*, vol. 81, no. 2, pp. 131–150, 1990.

[166] I. Peterlík, M. Sedef, C. Basdogan, and L. Matyska, "Real-time visio-haptic interaction with static soft tissue models having geometric and material nonlinearity," *Computers & Graphics*, vol. 34, no. 1, pp. 43–54, 2010.

[167] D. C. Brewstera, J. L. Cronenwett, J. W. Hallett, K. W. Johnston, W. C. Krupski, and J. S. Matsumura, "Guidelines for the treatment of abdominal aortic aneurysms: report of a subcommittee of the joint council of the american association for vascular surgery and society for vascular surgery," *Journal of vascular surgery*, vol. 37, no. 5, pp. 1106–1117, 2003.

[168] J. J. Monaghan, "Sph without a tensile instability," *Journal of Computational Physics*, vol. 159, no. 2, pp. 290–311, 2000.

[169] J. Swegle, D. Hicks, and S. Attaway, "Smoothed particle hydrodynamics stability analysis," *Journal of computational physics*, vol. 116, no. 1, pp. 123–134, 1995.

[170] J. P. Morris, *Analysis of smoothed particle hydrodynamics with applications*. Monash University Australia, 1996.

[171] D. Mandell, C. Wingate, and L. Schwalbe, "Computational brittle fracture using smooth particle hydrodynamics," tech. rep., Los Alamos National Lab., NM (United States), 1996.

[172] J. Chen and J. Beraun, "A generalized smoothed particle hydrodynamics method for nonlinear dynamic problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 190, no. 1, pp. 225–239, 2000.

[173] M. Liu, W. Xie, and G. Liu, “Modeling incompressible flows using a finite particle method,” *Appllied Math Model*, vol. 29, no. 12, p. 1252–1270, 2005.

[174] J. P. Morris, P. J. Fox, and Y. Zhu, “Modeling low reynolds number incompressible flows using sph,” *Journal of computational physics*, vol. 136, no. 1, pp. 214–226, 1997.

[175] C. Hirsch, *Numerical Computation of Internal and External Flows: The Fundamentals of Computational Fluid Dynamics: The Fundamentals of Computational Fluid Dynamics*, vol. 1. Butterworth-Heinemann, 2007.

[176] C. Fletcher, *Computational techniques for fluid dynamics 2: Specific techniques for different flow categories*. Springer Science & Business Media, 2012.

[177] S. Große and W. Schröder, “Two-dimensional visualization of turbulent wall shear stress using micropillars,” *AIAA journal*, vol. 47, no. 2, pp. 314–321, 2009.

[178] S. P. Sutera and R. Skalak, “The history of poiseuille’s law,” *Annual Review of Fluid Mechanics*, vol. 25, no. 1, pp. 1–20, 1993.

[179] J. Krog, B. Folkow, R. Fox, and K. L. Andersen, “Hand circulation in the cold of lapps and north norwegian fishermen,” *Journal of applied physiology*, vol. 15, no. 4, pp. 654–658, 1960.

[180] P. Ruengsakulrach, N. Eizenberg, C. Fahrer, M. Fahrer, and B. F. Buxton, “Surgical implications of variations in hand collateral circulation: anatomy revisited,” *The Journal of thoracic and cardiovascular surgery*, vol. 122, no. 4, pp. 682–686, 2001.

[181] H. K. Gaggin and J. L. Januzzi Jr, *MGH Cardiology Board Review*. Springer Science & Business Media, 2013.

[182] R. E. McCarthy, J. P. Boehmer, R. H. Hruban, G. M. Hutchins, E. K. Kasper, J. M. Hare, and K. L. Baughman, “Long-term outcome of fulminant myocarditis as compared with acute (nonfulminant) myocarditis,” *New England Journal of Medicine*, vol. 342, no. 10, pp. 690–695, 2000.

[183] S. Li and W. K. Liu, “Meshfree and particle methods and their applications,” *Applied Mechanics Reviews*, vol. 55, no. 1, pp. 1–34, 2002.

[184] H. Nakamura, T. Hashimoto, H. Oi, and S. Sawada, “Transcatheter oily chemoembolization of hepatocellular carcinoma,” *Radiology*, vol. 170, pp. 783–786, 1989.

[185] L. Hernández, J. E. Juliá, S. Paranjape, T. Hibiki, and M. Ishii, “On the use of area-averaged void fraction and local bubble chord length entropies as two-phase flow regime indicators,” *Experiments in fluids*, vol. 49, no. 5, pp. 1147–1160, 2010.

[186] K. Perktold, R. Peter, and M. Resch, “Pulsatile non-newtonian blood flow simulation through a bifurcation with an aneurysm.,” *Biorheology*, vol. 26, no. 6, pp. 1011–1030, 1988.

[187] K. Perktold and G. Rappitsch, “Computer simulation of local blood flow and vessel mechanics in a compliant carotid artery bifurcation model,” *Journal of biomechanics*, vol. 28, no. 7, pp. 845–856, 1995.

[188] H. C. Hemker, E. Loelinger, and J. J. Veltkamp, *Human Blood Coagulation: Biochemistry, Clinical Investigation and Therapy*, vol. 1. Springer Science & Business Media, 2012.

[189] P. Y. Huang and J. D. Hellums, “Aggregation and disaggregation kinetics of human blood platelets: Part i. development and validation of a population balance method.,” *Biophysical journal*, vol. 65, no. 1, p. 334, 1993.

[190] V. Interrante and C. Grosch, “Strategies for effectively visualizing 3d flow with volume lic,” in *Proceedings of the 8th conference on Visualization’97*, pp. 421–ff, IEEE Computer Society Press, 1997.

[191] B. Cabral and L. C. Leedom, “Imaging vector fields using line integral convolution,” in *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pp. 263–270, ACM, 1993.

[192] G. V. Bancroft, F. J. Merritt, T. C. Plessel, P. G. Kelaita, R. K. McCabe, and A. Globus, “Fast: a multi-processed environment for visualization of computational fluid dynamics,” in *Proceedings of the 1st conference on Visualization’90*, pp. 14–27, IEEE Computer Society Press, 1990.

[193] W. J. Schroeder, K. M. Martin, and W. E. Lorensen, “The design and implementation of an object-oriented toolkit for 3d graphics and visualization,” in *Proceedings of the 7th conference on Visualization’96*, pp. 93–ff, IEEE Computer Society Press, 1996.

[194] A. Fuhrmann and E. Gröller, “Real-time techniques for 3d flow visualization,” in *Proceedings of the conference on Visualization’98*, pp. 305–312, IEEE Computer Society Press, 1998.

[195] M. Schirski, T. Kuhlen, M. Hopp, P. Adomeit, S. Pischinger, and C. Bischof, “Efficient visualization of large amounts of particle trajectories in virtual environments using virtual tubelets,” in *Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*, pp. 141–147, ACM, 2004.

[196] H. Pfister, M. Zwicker, J. Van Baar, and M. Gross, “Surfels: Surface elements as rendering primitives,” in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 335–342, ACM Press/Addison-Wesley Publishing Co., 2000.

[197] F. H. Post and J. J. van Wijk, “Visual representation of vector fields,” *Scientific Visualization: Advances and Challenges*, vol. 23, pp. 367–390, 1994.

[198] D. N. Ku, “Blood flow in arteries,” *Annual Review of Fluid Mechanics*, vol. 29, no. 1, pp. 399–434, 1997.

[199] L. S. Tong, *Boiling heat transfer and two-phase flow*. John Wiley and Sons, New York, 1964.

[200] G. W. Knight, J. C. Taylor, M. F. Clem, F. F. Eaves III, and A. B. Lumsden, “Method and devices for endoscopic vessel harvesting,” Sept. 16 1997. US Patent 5,667,480.

[201] T. Yoshifumi, M. Shigenobu, N. Kazuto, U. Setsuo, F. Masakazu, H. Minoru, and H. Tomio, “Mutagenicity of smoke condensates induced by co 2-laser irradiation and electrocauterization,” *Mutation Research/Genetic Toxicology*, vol. 89, no. 2, pp. 145–149, 1981.

[202] C. Hensman, D. Baty, R. Willis, and A. Cuschieri, “Chemical composition of smoke produced by high-frequency electrosurgery in a closed gaseous environment,” *Surgical endoscopy*, vol. 12, no. 8, pp. 1017–1019, 1998.

[203] B. L. Ziegler, C. A. Thomas, T. Meier, R. Müller, T. M. Fliedner, and L. Weber, “Generation of infectious retrovirus aerosol through medical laser irradiation,” *Lasers in surgery and medicine*, vol. 22, no. 1, pp. 37–41, 1998.

[204] A. Selle, N. Rasmussen, and R. Fedkiw, “A vortex particle method for smoke, water and explosions,” in *ACM Transactions on Graphics (TOG)*, vol. 24, pp. 910–914, ACM, 2005.

[205] S. He, H.-C. Wong, and U.-H. Wong, “An efficient adaptive vortex particle method for real-time smoke simulation,” in *Computer-Aided Design and Computer Graphics (CAD/Graphics), 2011 12th International Conference on*, pp. 317–324, IEEE, 2011.

Chapter 9

List of Publications

Jichuan Wu, Chee Kong Chui, P. Nguyen Binh, and Chee Leong Teo. "Real-time rendering of drug injection and interactive simulation of vessel deformation using GPU." In Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE (pp. 4569-4572), 2013.

Jichuan Wu, Chee Kong Chui, Chee Leong Teo, and Oon Cheong Ooi. "Simulation of Smoke During Endoscopic Vessel Harvesting Surgery." In 11th Anniversary Asian Conference on Computer Aided Surgery, 2015 (Best paper candidate).

Jichuan Wu, Chee Kong Chui, and Chee Leong Teo. "A Software Component Approach For GPU Accelerated Physics-based Blood Flow Simulation." In IEEE International Conference on Systems, Man, and Cybernetics, 2015.

Jichuan Wu, Xin Mao, and Songfeng Lu. "A Qualitative Performance Comparison and Analysis of Suffix Array,FM-index and Compressed Suffix Array." In 2012 International Conference on Future Information Technology and Management Science and Engineering (pp. 348-352). 2012.

Haifeng Li, Jichuan Wu, Jianbo Liu and Yubing liang. "Finite Element Mesh Generation and Decision Criteria of Mesh Quality." Journal of China Mechanical Engineering, vol.23, pp. 368-377, 2012.

Jichuan Wu, Chee Kong Chui, and Chee Leong Teo."Hemodynamic Simulation of Blockage of Blood Circulation using Meshless Method and Dynamic Medical Images." Submitted to Journal of Medical Engineering and Physics. 2015.